AMSTRAD CPC

AMSTRAD CENT POUR CENT

HITS
NIGEL MANSELL
INDIANA JONES IV

INCROYABLE
DES CENTAINES DE JEUX A ACHETER OU TELECHARGER

DOSSIER
LES MEILLEURES DEMOS

LISTING
LE RETOUR DES SEPT NAINS

CUSTOMISEZ VOTRE ASSEMBLEUR

ISSN 0988 8160 - 47 - 25 F - 180 FL - 183 FB - 1660 CFA



N° 47 25 F
TRIMESTRIEL
AVRIL 93

SOMMAIRE

EDITORIAL

Saviez-vous que vous tenez entre les mains un véritable bijou ?

En effet, vous avez été nombreux à nous réclamer le retour de la rubrique « les sept nains », de notre ami le Barbare. Courez donc voir les pages listing, car nous avons sélectionné le « best of » de cette magnifique rubrique qui fut, sans conteste, une des plus appréciées de nos lecteurs.

Et les démos ? Ça fait des années que vous nous réclamiez un dossier sur cette tendance. Voilà qui est fait, pour notre plus grand plaisir, par le grand Megadeth en personne.

Pour les mordus, ne manquez pas notre rubrique spéciale « petits prix » qui vous permettra de jouer des nuits durant à vos aventures favorites.

Bien sûr, toutes les rubriques qui vous comblent d'aise sont au rendez-vous, et n'oubliez surtout pas notre serveur 3615 ACPC, qui vous offre, comme d'habitude, une montagne d'informations et de jeux à télécharger tranquillement chez vous.

Il ne nous reste plus qu'à prendre rendez-vous au mois de juillet pour passer de bonnes vacances ensemble.

AMSTRAD CENT POUR CENT

AMSTRAD CENT POUR CENT est une publication de **MEDIA SYSTEME EDITION** 19, rue Louis-Pasteur, 92513 BOULOGNE CEDEX - Tél. : 41 10 20 30.

Directeur de la publication : Alain KAHN.

Rédacteur en chef : Alain MASSOUMIPOUR. **Secrétaire de rédaction :** Ivan GAUCHER.
Ont participé à ce numéro : Denis JARRIL, Les LOGON SYSTEM, Christophe JORGE, Xavier LAMBERT, LUDOTRONIC, Claude LESUEUR, MEGADETH, Valérie RICHARD (traduction). **Illustrateur :** MYKAÏA. **Secrétariat :** Valérie RICHARD. **Permanence téléphonique** tous les mercredis de 10 h à 18 h.

Publicité : tél. : 41 10 20 40. **Responsable de la publicité :** Perrine MICHAELIS. **Chef de publicité :** Nezhia BOUBEKRI.
Responsable marketing et télématique : Barbara RING-REMY. **Minitel :** 3615 ACPC.

Responsable de la fabrication : Isabelle DERVEAUX-BERTÉ. **Photogravure :** Pi-M. **Impression :** IEI Lisses.
Service abonnements : B.I.I. BP 11-60940 Monceaux - Tél. : (16) 44 72 77 55.

Média Système Edition, SA au capital de 250 000 F RCS Nanterre B341 547 024. **Président-directeur général :** Alain KAHN.
Responsable financière et comptable : Sylvie BRUTINAUD. **Comptabilité :** Françoise LE METAYER. **Administration :** Jacqueline SEVASTELLIS. **Gestion :** Fabrice MORGAUT.
Commission paritaire en cours - Distribution NMPP - Dépôt légal janvier 1993.

Couverture : NIGEL MANSELL'S WORLD CHAMPION SHIP.

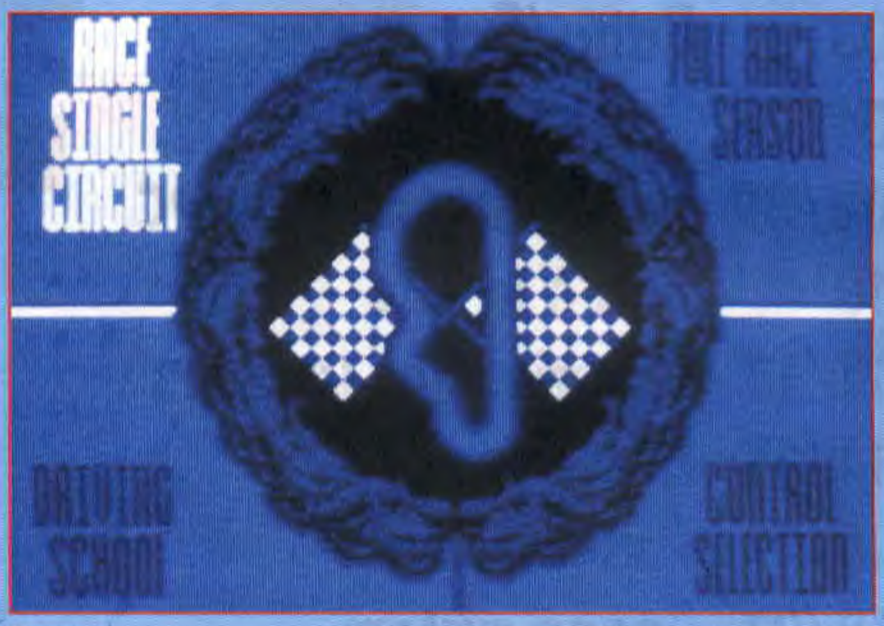
NIGEL MANSELL'S WORLD CHAMPIONSHIP

Voici enfin l'adaptation CPC de ce jeu désormais célèbre sur d'autres supports. Pas besoin de vous présenter le champion du monde de Formule 1, Nigel Mansell, je pense que vous avez entendu parler de lui, sinon c'est à désespérer de tout.

A l'heure où notre Alain Prost national vient de remporter la première course de la saison, à Kyalami, nous sommes en direct de la rédaction de *Cent Pour Cent* pour vous rendre compte des premiers essais de Nigel Mansell's World Championship.

NIGEL MANSELL EN BREF

Longtemps boudé par la chance, Nigel Mansell a toujours fait preuve d'un grand talent et d'une grande persévérance. En 1978, il va même jusqu'à revendre ses biens personnels et sa maison pour payer sa participation à la saison de Formule Ford. Ce n'est qu'en 1981 qu'il commence à récolter quelques menus succès avec l'écurie Lotus. En 1991, après un court séjour au sein de l'écurie Ferrari, Nigel Mansell revient dans l'écurie Williams, et l'année se solde par une seconde place au championnat du monde. Et s'il rate le titre, c'est à cause d'un incroyable manque de chance. Enfin, en 1992, il vit la récompense de ses efforts et de son talent, avec une invraissemblable succession de victoires et de records, plus le titre de champion du monde à la clef. Cette année 1992, qui fut certainement le couronnement de la carrière de Nigel Mansell, vous allez la revivre grâce à cette simulation offerte par Gremlin.



Le menu.

UN CHAMPIONNAT AU CHOIX

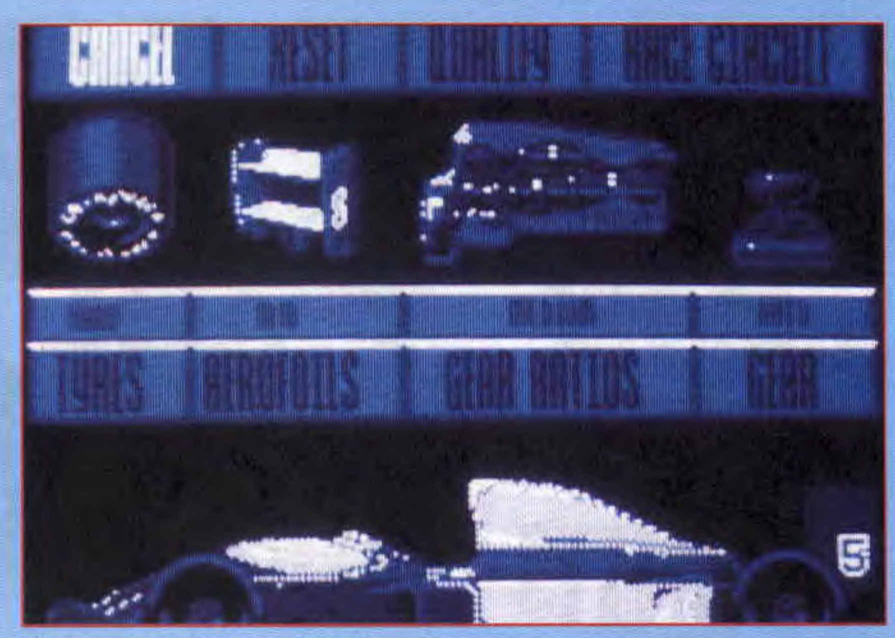
Vous allez pouvoir disputer le championnat du monde ou participer à des courses, au hasard de la saison. Vous piloterez bien sûr sur les seize circuits officiels de la FISA (Fédération internationale des sports automobiles). Votre but sera de placer Nigel Mansell sur la première place du podium et ce aussi souvent que possible. Etes-vous prêt à relever le défi ?

Après le chargement de l'écran principal du jeu, vous aurez la possibilité de choisir entre les options suivantes : Race Single Circuit (choisissez une seule course), Full Race Season (une saison complète de F1), Driving School (école de conduite) et Control Selection (sélection du système de contrôle). Vous pourrez avec cette dernière option paramétrer le système de contrôle que vous préférez. Vous pouvez jouer au joystick ou au clavier (avec les touches A, Q, O, P et la barre d'espace) ou encore modifier le jeu de touches que vous désirez utiliser.



Le Grand Prix de France sur le circuit de Magny-Cours.

Grâce à l'option école de conduite, vous allez, au fur et à mesure, apprendre à piloter votre voiture sur les différents circuits du monde. Notons à cette occasion la superbe idée utilisée par Gremlin de faire progresser le joueur en fonction de ses résultats. Par exemple, vous êtes sur le circuit de Magny-Cours et la trajectoire idéale est indiquée sur la piste. Si vous terminez le tour en moins de temps que ne vous l'indique le programme, vous disposerez d'une vitesse supplémentaire sur votre boîte. Vous passerez alors à la troisième, puis la quatrième et ainsi de suite jusqu'à pleinement maîtriser le circuit et disposer des six vitesses de la voiture. Si vous êtes prêt pour le championnat, il est temps de quitter l'école de conduite et de passer aux choses sérieuses.



Choisissez avec soin vos équipements.

CONFIGURER SA VOITURE

Avant de prendre le volant et de vous lancer à corps perdu dans la course, il est conseillé de faire quelques tours de circuit à l'occasion d'une course hors championnat et d'étudier les réactions de la voiture avec différentes configurations. Par exemple, sur un circuit rapide, il est fortement conseillé d'utiliser des pneus tendres et de changer les ailerons de manière à obtenir une meilleure vitesse de pointe. Les boîtes de vitesses, disposant de rapports plus ou moins longs, vous permettront d'adapter au mieux la voiture à votre conduite et au circuit. Une fois le bolide paré, il faudra, comme dans la réalité, vous qualifier pour prendre place sur la grille de départ. Chaque grille comporte douze places et vous pourrez vous trouver sur n'importe laquelle de ces douze places. Une fois qualifié, vous participez à la course et vous voyez à la fin le nombre de points marqués pour le championnat. Les points sont attribués comme dans la réalité, c'est-à-dire de la première à la sixième place : 10, 6, 4, 3, 2, 1. Si vous terminez dans les trois premiers, un écran apparaît vous montrant le podium de la course.

BILAN GLOBALEMENT POSITIF

Même si Nigel Mansell's World Championship ne fait pas partie des meilleurs logiciels de simulation de course, il n'en reste pas moins un bon petit logiciel en cette année 93, qui s'avère, hélas ! nettement moins fournie que les précédentes. Les graphismes, quoique peu colorés, sont d'une bonne qualité et le jeu est agréable tant au clavier qu'au joystick. Il aurait fallu que le produit soit un peu moins facile, mais vous pourrez corser l'affaire en changeant la boîte automatique par défaut en boîte manuelle. La bande sonore est tout ce qu'il y a de plus simple et un effort aurait sans doute pu être fait à ce niveau.

Finalement, on peu dire que Nigel Mansell's World Championship est un logiciel de bonne facture, mais qu'il aurait pu être de bien meilleure qualité en d'autres temps. A l'heure où blanchit la campagne (pardon chef de reprendre tes dires), je m'en vais faire un petit tour à Magny-Cours pour me dérouiller le poignet... A tout à l'heure pour la course.



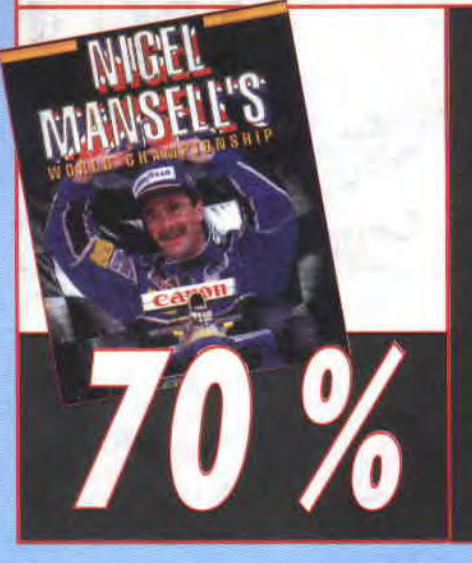
C'est parti... roulez petits bolides.

Racing Bull



Nigel Mansell's World Championship is the last products from Gremlin for the CPC computer. It's not the best driving simulation the CPC have ever had but it's not the badest one. The game is very simple, perhaps sometimes too simple to play but it has a lot of option that made it a good game for the CPC. Make us get some fun on this driving simulation, perhaps one of the last on our poor CPC. Thank you Gremlin to remember that the CPC is already on the course !

NIGEL MANSELL'S de GREMLIN



Graphisme :	70%
Son :	60%
Animation :	80%
Richesse :	90%
Scénario :	55%
Ergonomie :	65%
Notice :	90%
Longévité :	60%
Rhaal/Lovely :	65%

INDIANA JONES AND THE FATE OF ATLANTIS

Le re-voilà, armé de son éternel fouet, et comme toujours, accompagné d'une charmante jeune femme, Indiana Jones est de retour.



On vous présente l'histoire.

Eh oui ! il revient, mais, chose bizarre et inattendue, le soft sort avant le film (film qui, d'ailleurs, ne doit pas voir le jour sur les écrans). M'enfin, ce n'est là qu'un détail, car la version action d'Indy IV, après son succès sur les autres machines, se devait de sortir sur nos bons vieux CPC chéris. Montez au grenier récupérer votre fouet et votre vieux feutre, une aventure pleine d'embûches vous attend.

L'HISTOIRE

L'action se situe en 1938, le Dr Jones revient tout juste de sa dernière aventure avec deux objets, ô combien étranges, puisqu'il s'agit d'une petite statuette représentant un minotaure et une perle scintillante nommée Orichalque. Ces deux objets d'apparence inutile sont non seulement la preuve que la mystérieuse cité d'Atlantis a réellement existé, mais ils permettent d'envisager qu'il est encore possible de la retrouver. Tout ceci formerait une quête bien simple et bien banale aux yeux de notre archéologue zoophobe préféré, si les nazis n'avaient pas décidé, eux aussi, de retrouver Atlantis.



L'aventure peut commencer...

MAIS POURQUOI LES NAZIS ?

Chose étrange, une fois de plus les nazis viennent fourrer leur gros nez dans les affaires d'Indy. Mais cette fois-ci, l'enjeu est énorme, puisque ces gros vilains pourraient trouver, avec Atlantis, une mine quasi inépuisable d'Orichalque, billes extrêmement précieuses puisqu'elles dégagent une énergie hors du commun. Imaginez un peu les armes destructrices que ces méchants nazis utiliseraient sans vergogne. De plus d'après le dernier livre de Platon (livre qui décrit amplement la cité), Atlantis pourrait également transformer de simples humains, comme vous et Francky (euh, non, pas Francky), de simples mortels comme vous et moi, en créatures surpuissantes puisqu'il s'agit de dieux. Eh oui, voyez déjà toutes les choses horribles que ces affreux nazis pourraient faire. Pour Indy, il n'est pas question de les laisser trouver la cité, alors à vos fouets et à vos écrans, le sort de l'humanité en dépend...

LE JEU !

Bon, OK, je vous parle du jeu proprement dit, ce dernier se décompose en 6 niveaux ! Le casino, d'abord, où vous devrez trouver Alain Trottier et marchander au meilleur prix les objets et les armes qu'il vous proposera. Malheureusement, les nazis arrivent, vite il faut fuir pour arriver à la base navale, où, après avoir évité les multiples pièges, vous trouverez

le code secret qui vous mènera directement à la base sous-marine. Ici point de combats, mais de la stratégie et de la réflexion (la grue s'avérera le bon moyen pour passer). Ensuite, le sous-marin vous attend, avec ses bombes meurtrières. Indy devra montrer ici ses talents de démineur (profitez-en pour vous venger de l'infâme Klaus Kerner). On arrive au bout, l'archipel est en vue. Il faudra trouver la bonne île (grâce aux conseils de l'homme dans la caverne) pour arriver enfin, non pas au bout de vos peines, mais au commencement : Atlantis. Les nazis, toujours aussi pots de colle, vous y attendent, les créatures surnaturelles et les mécanismes vicieux seront votre seul espoir.

DES OPTIONS SYMPA

Ce jeu possède de nombreuses facettes, certaines sont assez cool. Vous pourrez gérer votre personnage avec le clavier ou le joystick. Stop, je viens de dire une bêtise, il ne s'agit pas d'un personnage, mais de deux : Indy et Sophia, que vous pouvez incarner à tour de rôle. De plus, vous pouvez choisir votre angle de vue (les quatre points cardinaux), ainsi, d'un même endroit, il sera plus aisé de trouver les objets ou de déceler les passages secrets ...

POUR FINIR

L'adaptation de ce jeu sur CPC, n'aura tout de même pas été une chose grandiose, les déplacements sont lents et les graphismes assez fades, m'enfin, les grandes caractéristiques d'Indy ont été conservées, si bien que le jeu est intéressant ! Compatible pour toute machine avec lecteur disquette, et en plus la notice est en français.

Indiana Totov





Indiana Jones is back ! Eventhough moves are slow and graphisms are dull, sort of a plot gives this game a good interest. You can choose your character, Indy or Sophia, and also choose the visual angle which is more convenient...



Vous pourrez incarner aussi bien Indy...



... que Sophia, cool non ?

INDIANA JONES AND THE FATE OF ATLANTIS de LUCAS ARTS/LUCASFILM GAMES



Graphisme :	70%
Son :	-%
Animation :	73%
Richesse :	72%
Scénario :	99%
Ergonomie :	75%
Notice :	68%
Longévité :	79%
Rhaal/Lovely :	73%

BOOGIE WOOGIE

Il y a quelques années, sortait sur nos CPC un jeu nommé Pingoo. Aujourd'hui, Esat revient avec une nouvelle version : Boogie Woogie, qui, malgré de petites imperfections, reste un jeu sympathique.



Ce cœur restera de glace.

Vous incarnez Boogie, un petit pingouin bleu, et Woogie, un petit éléphant ro... euh non, un petit pingouin rose. Ces derniers, après le cambriolage de leur igloo, doivent retrouver leur fortune, éparpillée sur la banquise par les voleurs pressés. Cette fortune est uniquement composée de diamants bleus pour Boogie, et roses pour Woogie. Comment récupérer les bijoux ? Eh bien ! il n'y a que deux moyens : le premier consiste à attendre un hypothétique bonus, le second, à aligner les trois diamants de chaque tableau. Quant à vos adversaires phoques, Esquimaux ou requins, ils vous tueront si vous touchez l'un d'eux.

LE JEU !

Oui, j'y arrive. La présentation est sympa : un écran reformaté en overscan et des couleurs bien choisies. Les sprites sont amusants et l'animation est du plus bel effet, mais il faut noter une petite imprécision dans les contacts avec les ennemis. Par contre, un point fort, les bonus apparaissent au bon moment.

Le bon point : ce jeu est composé de 50 tableaux, autant dire que vous n'êtes pas prêt de le finir. Le mauvais point : la gestion du clavier est nulle. Jugez un peu : S (haut), F (droite) D (bas) A (gauche). Ce n'est vraiment pas pratique. Pour le deuxième joueur, tout rentre dans l'ordre, grâce au pavé numérique.

Dans le mode deux joueurs, le jeu est identique, chaque joueur vaque à ses affaires. La réussite de l'un d'eux entraîne automatiquement le passage au niveau supérieur pour les deux. On se marre bien, et on ne se mélange que très peu les pinceaux. Mais ça ne vaut pas Pang.

Grâce à l'éditeur, vous pourrez créer vos propres tableaux et mettre les monstres, les diamants et les joueurs où vous voulez.

C'EST FINI COMME LA MOQUETTE

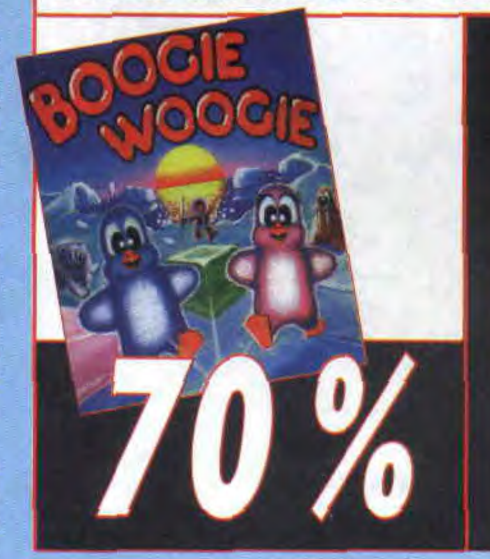
En conclusion, Boogie Woogie est un bon jeu pour nos CPC, d'autant que les programmeurs ont utilisé quelques trucs de démomaker ce qui laisse présager une nouvelle race de jeux. Comme toujours, quelques petites imperfections sont à regretter. Allez, c'est la vie, ils feront mieux la prochaine fois. Une bonne nouvelle pour votre petite sœur,ce jeu est accessible à partir de 5 ans.

Totov



Despite some defects, Boogie Woogie belongs to a news group of games. We can easily notice the originators are programmers with demomakers tendencies. But we're still far away from the CPC's limits, particularly for sprites management.

BOOGIE WOOGIE d'EAST SOFTWARE compatible CPC +



Graphisme :	80%
Son :	55%
Animation :	60%
Richesse :	55 %
Scénario :	70%
Ergonomie :	60%
Notice :	80%
Longévité :	90%
Rhaal/Lovely :	75%

TOUJOURS PRESENT

Vous avez eu peur de ne plus nous retrouver. Hein ? Ne dites pas non. Je l'ai lu dans toutes vos lettres humides d'émotion. Eh bien on est toujours là et, qui plus est, pour un bon moment.



COURRIER

Bon, passons tout de suite à votre courrier, car comme d'hab' il est conséquent.

Salut Francky,
Je t'écris pour savoir comment on tape les caractères bizarres qu'il y a dans le listing du CPC n° 45, LESMINGS ? Et bravo pour votre super canard.

Christophe H., Yannick et plein d'autres

Le caractère dont tu parles s'appelle l'arobace. C'est un caractère américain qui correspond à notre « à ». Pour des raisons techniques liées à la PAO (publication assistée par ordinateur), nous n'avons pas pu correctement éditer ce caractère. La rédaction tient donc à présenter ses plus plates excuses à tous ses lecteurs, ainsi qu'à l'auteur du listing qui ignorait que nous puissions rencontrer ce type de problème. Or donc, l'arobace s'obtient en appuyant sur la touche où se trouve un « à » accent grave sur un clavier AZERTY, alors que sur le clavier QWERTY c'est un tout petit « a » entouré d'un cercle (l'arobace américain). Dans le programme incriminé, ce caractère est utilisé pour représenter une partie du décor du jeu. En Basic, c'est le pointeur de variable. Il sert à connaître l'adresse d'une variable dans la mémoire du CPC. Il est rarement utilisé en Basic, mais lorsqu'on utilise des RSXs, il retrouve toujours sa place.

Dear Franck, I've a terrible problem Well, j'ai fait des redéfinitions de caractères, mais quand je veux les afficher, seul le dernier caractère est correctement modifié. Est-ce une erreur de ma part ou un big bug du Basic ? Bref, comment cela se fait-il ? Autre chose, comment faire pour programmer un minitraitement de texte sans utiliser la commande Input en Basic ?

Anne Onyme

A la vue du petit programme qui accompagnait ta lettre, je remarque que le manuel de l'utilisateur du CPC a encore frappé. Alors, mes bien chers frères, tous ensemble, reprenons en cœur la très sainte règle du SYMBOL AFTER et du MEMORY : LSANDEUQUSFDUPSIYAUMDLPIDEPDLSA, en d'autres mots, le SYMBOL AFTER ne doit être utilisé qu'une seule fois dans un programme. S'il y a un MEMORY dans ce dernier, il doit être placé avant le SYMBOL AFTER. Si vous ne respectez pas cette règle, vous vous exposerez aux foudres du très puissant Z80, notre maître à tous.
Pour ton traitement de texte, eh bien, tu devras utiliser l'instruction INKEY$ du Basic, et reprogrammer toutes les fonctions de saisie, de déplacement du curseur et d'effacement de caractère en milieu de ligne. Faute de place, je ne peux pas te dire ici comment faire, mais sois bien sûr que tu vas avoir du boulot.
Dans le courrier du numéro 46, j'ai omis de fournir la précision suivante : les imprimantes que l'on peut connecter sur un CPC doivent pouvoir émuler le standard EPSON FX, sinon ça risque de planter lamentablement. Ceci étant, passons à la suite...

Salut mon cher Francky,
Peut-on mettre un lecteur 3"1/2 prévu pour PC ou Amiga sur un CPC 6128 ? Est-ce que les disquettes peuvent se détériorer avec le temps ?

Zip

Mon cher Zip, on peut effectivement mettre un lecteur 3"1/2 sur un CPC, mais cela pose un certain nombre de problèmes, car il faut modifier le lecteur 3"1/2 pour qu'il devienne « simple face », ainsi le CPC pourra le gérer convenablement, sans utilitaire dans la mémoire de l'ordinateur. Néanmoins, le CPC ne pourra gérer qu'un total de 180 Ko par face. Il existe toutefois des lecteurs 3"1/2 pour CPC chez Jessico. Ceux-ci permettent de s'affranchir de cette barre des 180 Ko, mais leur prix reste supérieur à celui d'un lecteur 3"1/2 « modifié maison ».
Si on prend bien soin des disquettes, comme je vous l'ai déjà expliqué dans un précédent numéro, elles ne devraient pas se détériorer. Les disquettes ne sont pas biodégradables. Par contre, le lecteur, lui, peut s'user avec le temps. Qui n'a jamais eu sa courroie d'entraînement détendue ? Si ce cas de figure vous arrive, demandez une courroie neuve chez un accessoiriste et faites le remplacement vous-même, sinon le réparateur changera tout le lecteur, et vous vous en sortirez avec huit cents balles dans les gencives.

Salut Francky,
J'ai un petit problème avec Advenced OCP Art Studio. Je sauvegarde un dessin en mode 0, et quand je le rappelle en Basic, les couleurs ne sont pas les mêmes que sous OCP. Y a-t-il un programme rectifiant les couleurs ? Si oui, lequel ?

Senseby le magnifique

Mon cher Senseby, il existe effectivement un programme – à base d'RSX – qui s'occupe de charger les images d'OCP avec leurs vraies couleurs, et avec le cycling des couleurs, s'il vous plaît. C'est une œuvre de notre bon Olaf. Le listing est paru dans le numéro 40 de votre revue chérie, page 54, colonne du milieu. Il est également disponible sur notre serveur. Voilà une réponse précise. Non ?

Cher Maître Franck,
Salut ! Je croyais tout savoir sur l'Amstrad, mais il faut bien avouer que tu me dépasses. Voici donc quelques questions auxquelles j'aimerais que tu répondes. Après avoir vécu depuis 1985 avec un 464, j'ai finalement acheté un 6128. Est-il possible d'y connecter, en deuxième lecteur de disquette, celui du 464 ?
Peut-on transformer un CPC AZERTY en un CPC QWERTY ? Pourquoi ne mettez-vous pas un bon assembleur sur le Minitel ? Comment écrire un octet à un endroit spécifique d'une disquette ? Merci pour votre superbe journal et, surtout, n'arrêtez pas de bosser.

Nick

Il est parfaitement possible de connecter le lecteur de disquette d'un 464 sur un 6128, car c'est le même lecteur sur les deux machines. La seule différence se situe dans le câble de liaison. Sur le 464, il y a une grosse boîte avec le câble, c'est le contrôleur de disquette. Le 6128, lui, a déjà un contrôleur de disquette sur la carte mère (pour contrôler le lecteur interne). Ce contrôleur peut piloter (sans modifications) deux lecteurs. Tant et si bien que le 6128 n'a pas besoin de cette grosse boîte. Il vous suffit donc d'acheter un câble en nappe pour raccorder le lecteur du 464 au port « disc drive 2 » du 6128.
Il est impossible de transformer réellement un AZERTY en QWERTY et inversement, car la distinction ne réside pas uniquement dans la différence d'emplacement des touches. Dans la Rom, il y a une table avec les correspondances entre la position d'une touche et le caractère qui y est associé sur le clavier. Et comme une Rom ne se démonte pas avec un couteau suisse, on ne peut pas la changer aussi facilement qu'une touche. Donc, c'est impossible (toujours aussi précis). On ne met pas d'assembleur sur Minitel, car les distributeurs se réservent le droit de vente de ce type de logiciel. Donc... Pour finir et, comme vous le savez sûrement, une disquette est constituée de pistes qui contiennent des secteurs qui contiennent des octets. Seuls les secteurs peuvent être manipulés en lecture ou en écriture. Donc, si vous voulez changer un octet, il faudra charger le secteur qui le contient, modifier l'octet, puis sauvegarder le tout.

Salut Francky,
Cela fait bientôt deux ans que je suis ta rubrique, mais quelques lacunes subsistent encore dans mon éducation. Qu'est-ce que la pile ? et quelle est son utilité ? D'autre part, une rubrique « fer à souder » serait sympa, car j'aimerais gonfler mon CPC, et les extensions du commerce sont un peu chères à mon goût. Poum, dans son infinie bonté, nous a donné différentes manières d'afficher un sprite. Mais, facétieux comme il est, il ne nous a pas dit comment les effacer ?

Hugues

Imaginons une pile d'assiettes. Quand on en pose une, c'est obligatoirement sur le sommet, et quand on en enlève une c'est aussi sur le sommet. Donc : dernière arrivée, première partie. Ça, c'est le principe de la pile. Dans un programme, la pile est utilisée quand vous faites des GOSUB (ou des CALL en LM) vers un sous-programme. La particula-

rité d'un sous-programme, c'est qu'il fait un retour à l'appeleur dès qu'il trouve un RETURN (RET en LM). Mais pour qu'il puisse faire ce retour, il faut qu'il sache qui l'a appelé. Pour ça, lors d'un GOSUB, l'adresse de l'instruction suivant le GOSUB est posée sur la pile, et le sous-programme est lancé. Lors d'un RETURN, l'adresse qu'il y a en haut de la pile est retirée et le programme poursuit à cette adresse le cours du programme, donc juste derrière le GOSUB.
Pour être encore plus concret, la pile n'est qu'une zone mémoire de votre machine qui possède son propre pointeur (SP) qui n'est autre qu'un registre comme HL, BC, DE. Concernant le fer à souder, je vous réserve une surprise dans le prochain numéro. Pour effacer un sprite, c'est super simple, il suffit de mémoriser la portion de l'écran que le sprite va écraser avant de s'afficher, et quand on veut effacer le sprite, il suffit simplement d'afficher ce qu'on a mémorisé. Simple, non ? Une autre méthode fait appel aux routines d'affichage, utilisant le mode XOR. En effet, dans ce mode, afficher une deuxième fois un sprite revient à effacer ce dernier.

Salut Francky
J'ai un CPC 6128+ et j'ai acheté une DMP 3160 Amstrad il y a déjà près de six mois. Or je n'arrive pas à imprimer quoi que se soit. Pourtant, dans la documentation de l'imprimante, il est dit de taper LPRINT "bonjour" pour que...

Stoooooooop ! Nous avons ici un des plus beaux exemples de débutant croyant naïvement que la documentation d'un appareil est un livre saint ! Depuis quand une doc d'imprimante est un manuel de programmation ? Les documentations n'existent que pour vous donner la signification des boutons, des branchements, voire même des codes de contrôle, ou tout autre spécificité de l'imprimante, mais quand on programme une imprimante, on passe par l'ordinateur. C'est le manuel de l'ordinateur qu'il faut lire pour savoir comment on fait pour envoyer un caractère à l'imprimante. Sur un CPC il faut utiliser PRINT #8,"bonjour" et ce quelque soit le type d'imprimante connectée. Allez, pour frimer, sachez que LPRINT est l'équivalent PC de notre PRINT #8.

Hello Francky,
Sans plus attendre, je te pose ma question : A quoi sert le CP/M ?
L'ours blanc

Le CP/M est un très vieux système d'exploitation créé par Digital Research pour les machines à base de 8080 d'Intel (machines considérées comme des monstres de puissance, avec une fréquence d'horloge qui frisait le 1 MHz et une mémoire d'une trentaine de Ko). Il est inutile de vous dire que ce système est complètement obsolète, vu son âge et les utilitaires qu'il y a sur le CPC. Il n'empêche qu'en son temps, c'était quand même un bon petit système d'exploitation, mais bon, c'était une autre époque...

Salut Francky,
Os court, j'ai un méga problème avec le bankmanger et mes RSX : ils se font la guerre. Ils ne veulent pas fonctionner plus d'une seule fois ensemble. Pourquoi ?
Big Man

Tiens, c'est la première fois qu'on me rapporte ce genre de plantage. Après une très courte enquête, l'inspecteur « dirty » Francky trouva LA solution (lorsqu'il eut fini de dégommer quelques dizaines de bugs avec son 357 Magnum) : il ne faut initialiser qu'une seule fois une RSX, sinon elle risque de devenir instable et de planter le système. Lors de vos essais de débugage, mettez un Rem au début de la ou des ligne(s) qui charge(nt) et initialise(nt) vos RSX (une fois qu'ils auront été initialisés, bien sûr !). Ou faites un test du genre :

```
IF PEEK (&adresse)<>octetfichierRSX THEN
LOAD nom du fichier RSX$,&adresse:CALL &adresse
```

C'est simple, rapide, efficace et sans risque de plantage ! Merci qui ?

Salut Franck
Comment fait-on pour sauvegarder un fichier en binaire ? Comment fait-on pour rendre des fichiers visibles-invisibles ? Comment fait-on pour rendre ineffaçables des fichiers à lecture protégée Et tout ça avec Disco.
D. Jeko

Pour sauvegarder du binaire, il faut utiliser le célèble SAVE, mais avec trois paramètres :
SAVE nomdufichier$,B,&adressededébut,&longueur
Le B c'est pour indiquer que là on fait une sauvegarde en binaire d'une zone mémoire. &adressededébut, c'est l'adresse du début de la zone à sauvegarder. Et &longueur, c'est le nombre d'octets qu'il faut sauvegarder. Pour recharger le fichier : LOAD nomdufichier$,&adresse. Ici, &adresse est l'adresse où le fichier se chargera en mémoire. Et tout ça en Basic.
Un troisième paramètre peut-être ajouté à la ligne de sauvegarde qui indiquerait l'adresse d'exécution du prog permettant ainsi de le lancer directement par la commande RUN.
La protection à l'écriture et l'invisibilité de fichier sont codées sur l'extension du nom de fichier dans le directory : si les bits 7 des deux premiers caractères de l'extension sont à zéro, le fichier est un fichier visible et effaçable, sinon il est invisible et/ou protégé en écriture (avec Disco, renomme les fichiers en leur ajoutant l'option [p] ou/et [s]).

Salut Franck,
Qu'est-ce qu'un fichier source ? Qu'est-ce qu'un fichier code ? Qu'est-ce qu'une constante ? A quoi servent les instructions ORG, ENT $, EQU, DEFM, DEFB, DEFW, DEFS, en assembleur ?

Un fichier source, c'est un programme qui n'en est pas encore un. Il n'est pas compilé. C'est le listing du programme quoi ! Sur un CPC, tous les programmes en BASIC sont des sources (.BAS .ASM .ASS et pour certain .TXT). Un fichier code, c'est un programme compilé, donc exécutable, du langage machine (.BIN sur CPC) !
Une constante, c'est presque une variable, sauf qu'on ne lui donne sa valeur qu'une seule fois dans le programme, et on ne la modifie pas après. Les constantes en Basic interprétées ne servent pas à grand-chose, donc sur CPC, elles n'existent pas, à part, quelques constantes système comme PI. Bon, passons à la suite : en fait, tes instructions ne sont pas des instructions proprement dites, car elles ne servent qu'au bon assemblage du code. Ce ne sont pas des instructions du Z80, mais des instructions de l'assembleur utilisé. ORG adresse, sert à dire à l'assembleur que le code doit être assemblé pour une exécution à partir de l'adresse précisée. ENT $, c'est le point d'entrée du programme (lors d'un RUN du fichier binaire, l'exécution ne commence pas obligatoirement au premier octet du programme, mais à ce point d'entrée). Les EQU doivent toujours être placés au début du source, ils servent à associer une valeur à une constante, ce qui permet d'avoir des paramètres facilement modifiables sur la totalité du source lors de la mise au point du programme. DEFM inclut la chaîne de caractères placés derrière dans le programme à la position du DEFM. DEFB fait la même chose que DEFM, mais pour un/des octet(s) plutôt que pour une chaîne de caractères. DEFW comme DEFB, mais avec des mots (2 octets) plutôt que des octets. DEFS, quant à lui, permet de créer des zones tampon de *x* octets avec la valeur 0 par défaut *(voir le récapitulatif)*.
Bien voilà, c'est fini pour ce numéro de printemps, mais le Francky version été vous réserve de bonnes surprises...

Francky, la créature aux deux cerveaux

RECAPITULATIF DES PSEUDO-INSTRUCTIONS

	ORG	#XXXX	adresse d'implantations du programme.
	ENT	$	début du programme.
var	EQU	123	var=123 dans tout le programme.
	DEFM	'bonjour'	incorporation de bonjour dans le code.
	DEFB	#10,#50	incorporation des 2 octets #10 et #50
	DEFW	#4000,#10	incorporation des 4 octets #00,#40,#10,#00
	DEFS	10,#FF	incorporation de 16 octets égaux à 255

LE TRI MARRANT

Après avoir passé en revue la gestion optimisée de la mémoire ainsi que le tri à bulles, nous en voici venu à l'analyse d'une méthode quelque peu plus puissante : le tri par insertion. Il va nous permettre de mieux aborder encore le phénomène d'optimisation de la gestion de la mémoire.



Avant tout, je voudrais vous faire part d'une remarque qui est la déduction de quelque dix ans de programmation. Lorsqu'on commence à programmer, on se confectionne des routines permettant de réaliser telle ou telle fonction. Malheureusement, on s'aperçoit vite que les applications réalisées ne se révèlent pas aussi puissantes et rapides que prévu. On a beau changer de langage pour aboutir à l'assembleur, rien n'y fait. Le déroulement de certaines tâches se révèle toujours manquer de peps, de tonus, bref, de vélocité. Pourtant, arrivé à ce point, on pourrait se dire que le temps machine est pleinement exploité... Mais il manque encore le petit rien qui fait qu'un programme bombarde réellement. Ce petit rien n'est ni plus ni moins qu'un gros paquet de matière grise qu'on n'a pas assez investi dans le produit. L'algorithmie, soit, en informatique, la science qui permet de trouver systématiquement la bonne technique de traitement en fonction des opérations à réaliser, est un point à ne délaisser sous aucun prétexte. Quel que soit le langage utilisé, on ne peut se permettre de laisser de côté la part de réflexion précédant toute écriture de la moindre ligne de code.

IL EST TEMPS DE REFLECHIR

Si nous avons perdu quelques lignes pour exprimer une telle idée, c'est qu'elle est très importante. La bonne méthode de programmation rendra le code bien plus efficace et permettra d'obtenir des vitesses de traitement correctes, même en Basic. En voici une preuve irréfutable : les tests sur les tris. Nous avons vu la première méthode, bête et méchante. Désagréablement systématique. Nous allons maintenant passer en revue la seconde, plus subtile, qui est le tri par insertion. Nous finirons par l'essence du tri : le tri rapide. Notez qu'entre ces algorithmes, il existe des rapports de puissance qui ne sont pas à négliger. Le tri par insertion est quelque 4 fois plus rapide que le tri à bulles qui est 70 fois plus lent que le tri rapide. Comme vous le voyez, il faut parfois se gratter quelque peu les méninges pour utiliser 1/70e du temps normalement gaspillé par des routines de premier jet. Ayons une pensée pour les maîtres de la programmation.

INSEREZ LES RANGS

Le principe du tri par insertion est simple comme bonjour. Il consiste simplement, à partir d'une zone triée, à insérer les éléments à leur place dans le tableau grandissant. Imaginons que nous voulions trier un tableau d'entiers dans l'ordre croissant. Nous partons du second élément que nous stockons dans une variable temporaire. Ensuite, nous testons cette valeur à la valeur contenue dans la cellule précédente du tableau. Si cette dernière est supérieure à la temporaire, nous la décalons vers le haut. Comme nous sommes arrivés en début de table, nous recommençons à partir du troisième élément que nous stockons dans une variable temporaire. Tant que celle-ci est supérieure aux valeurs trouvées en descendant dans le tableau, nous descendons. Lorsque la variable temporaire est plus grande que la valeur trouvée dans la celulle du tableau, nous rangeons la temporaire dans l'entrée juste au-dessus. Puis nous reprenons ensuite à la celulle suivante, soit la quatrième...

DE LANGAGE EN LANGAGE

Avant de se lancer dans l'écriture proprement dite de la routine, nous

allons décrire les opérations à réaliser dans un basicois quelque peu franchouillard. En voici le pseudo listing dans lequel Tab est le tableau à trier, et Temp la variable temporaire.

```
Pour i allant de 2 à la taille de Tab
     Temp = Tab(i)
     j = i-1
     Tant que j > 0 et que Tab(j) > Temp
          Tab(j+1) = Tab(j)
          j = j - 1
     Fin du tant que
     Tab(j+1) = Temp
Suivant
```

Si vous imaginez le déroulement de cette routine, vous vous apercevrez très vite que ce programme n'est absolument pas sorcier. Comme le veut la règle, il met en œuvre quatre fois moins d'échanges que le tri à bulles, ce qui n'est pas peu dire.

TROIS TABLEAUX POUR MIEUX POINTER

Comme le mois dernier, nous n'allons pas nous amuser. Si nous avons à trier un tableau de chaînes, nous ne déplacerons pas ces gros paquets de caractères. Mieux vaut passer par un tableau de pointeurs qui permettent de déplacer seulement deux octets plutôt que *n* octets formant la chaîne effective. Pour cela, nous utiliserons un second tableau, d'entiers, dont les entrées contiennent les indices des chaînes du second tableau qui, lui, ne bouge jamais. Cela permet non seulement de gagner du temps sur les mouvements de mémoire, mais aussi sur le « garbage collection ». Ce terme barbare identifie simplement l'action que fait le système lorsqu'il réorganise sa mémoire. Lorsque vous allouez des chaînes de caractères, le Basic prend des places çà et là dans la mémoire. Si vous détruisez, réduisez ou agrandissez des chaînes de caractères, la mémoire se fragmente en alternant des zones occupées et des zones libres. Si les zones libres se révèlent être trop petites pour être utilisables directement, le Basic lance un « garbage collection ». Dans cette opération, il rassemble toutes les zones utilisées d'un côté, et tous les espaces libres de l'autre, de manière à obtenir une grande plage de mémoire disponible.

QUATRE POINTS CRUXIAUX

Lorsque le Basic réalise cette opération, il arrive qu'il prenne énormément de temps. Si cela se passe lors d'un traitement sur chaîne déjà long, vous ne vous apercevrez pas de la supercherie. Vous serez alors simplement désolé que votre programme se traîne quelque peu à cet endroit. Pour éviter le « garbage collection », il faut simplement faire très attention à manipuler le moins possible les chaînes de caractères qui fragmentent énormément la mémoire. Rien de plus évident alors d'utiliser, si cela est possible, des pointeurs. Si cela n'est pas le cas, il faut forcer des allocations trop grandes. Ainsi, si vous définissez une chaîne par :

A$ = STRING$(" ", 255): LA% = 0

Cela signifie que la chaîne A$ prend 255 caractères en mémoire, mais que sa longueur théorique, stockeé dans la variable LA%, est de 0 octets utilisables selon votre protocole. Vous pouvez ainsi remplacer des caractères en son sein, sans modifier son emplacement mémoire et donc sans entraîner le moindre « garbage collection ». C'est assez contraignant à manipuler, gourmand en mémoire, mais le résultat est probant et la vitesse non perdue peu négligeable, si vous traitez de nombreuses chaînes de caractères. Dans notre cas, nous utiliserons un tableau d'index.

CINQ LIGNES EFFICACES

Voici donc le listing de tri de chaînes de caractères que nous attendons tous avec impatience.

```
10 Dim Tab$(100)
20 Dim Index%(100)
30 For i%=1 to 100
35 Rem Initialisation de l'index
40   Index%(i%)=i%
50 Next i%
60 Rem Remplissage à votre gré de Tab()
61 Rem En lisant un fichier texte
62 Rem Et en le stockant dans Tab$(), par exemple
70 Rem Départ du tri
80 For i%=2 to 100
90   Temp%=Index%(i)
100  j%=i%-1
110  While (j%>0 and Tab$(Index%(j%))>Tab$(Temp%)
120       Index%(j%+1)=Index%(j%)
130       j%=j%-1
140  Wend
150  Index%(j%+1)=Temp%
160 Next
170 Rem Affichage du tableau trié
180 For i%=1 to 100
190  Print Tab$(Index%(i%))
200 Next
```

Comme nous le voyons, si la comparaison des chaînes de caractères porte bien sur le tableau Tab$, les déplacements de ces chaînes se font dans le tableau de pointeurs que nous avons appelé Index%. De surcroît, cette méthode permet de garder la trace de l'ordre d'affichage. Cet ordre, qu'on appelle physique, est dans certains cas de gestion très important. Il permet de retrouver des traces significatives dans la base de données.

SIX TABLEAUX POUR UN EMPIRE

Comme dans le dernier numéro, nous pouvons parfaitement utiliser ce type de tri sur des informations stockées dans plusieurs tableaux. Ainsi, nous rassemblons des données de chaque tableau de manière à former un enregistrement. Cela permet encore une fois de gérer différents ordres triés à partir d'un même ordre physique. Indexnom%() permet d'indexer les données selon le champ (le tableau) Nom$, Indexcp%() selon Codepostal$... Dans tous les cas, il nous faut programmer la méthode de tri correspondant à chaque index et un tableau d'index maitres, recopie de l'index en cours. Cette dernière astuce est gourmande en mémoire mais permet de ne recourir qu'à une routine d'acquisition de données ne passant que par l'index maître.

SEPT ASSEZ

Lorsqu'on programme un tri, il y a une chose importante à prendre en compte. Elle se nomme la stabilité du tri. Lorsque vous rangez vos informations selon des méthodes diverses, le fait que deux données de valeurs égales restent dans le même ordre avant et après tris permet de dire que le tri est stable, car il trie sans mélanger les informations. Si vos données se promènent selon le bon vouloir anarchique du trieur, nous nommerons notre tri instable. Comme une des méthodes permettant de réaliser des tris multicritères passe par un second tri d'une base de données prétriée, il faut absolument, dans ce cas, que la stabilité soit. Le tri par insertion, est stable. Donc, pas de problème de ce côté.

Attention ! préparez les combinaisons ignifugées. En effet, dans le prochain numéro, nous aborderons le tri rapide. De quoi décoiffer Telly Savalas.

Sined le Barbare à bière

L'ASSISTANT...

En assembleur, ce qui compte le plus c'est le cerveau du programmeur. Il va sans dire, d'ailleurs, qu'une part importante de la programmation passe par cet organe. Mais la puissance de l'assembleur ainsi que les méthodes à utiliser ne sont absolument pas à négliger...



Il est bien loin le temps où on programmait en assembleur avec des interrupteurs. C'est ainsi qu'on insérait des programmes au sein de la machine. On positionnait des commutateurs de manière à agir sur huit bits, et cela fait, on pressait sur un bouton de validation. Inutile de vous dire que si la moindre erreur se glissait dans un programme, il valait mieux tout recommencer à zéro. Au regard de ces méthodes archaïques, nos assembleurs sont des usines à gaz de grande puissance. Pourtant, si on regarde d'autres machines, comme l'Amiga ou le PC, on s'aperçoit que les outils dont elles disposent sont encore au-dessus des capacités de nos bons vieux camarades. DevPac II ou Masm sont de véritables monstres au regard de Dams ou de Pyradev. Mais entre tout cela, n'y a-t-il pas de méthodes nous permettant de choisir efficacement un outil selon le travail à effectuer ?

A BON TRAVAILLEUR, BONS OUTILS

Ne connaissez-vous pas ce bon vieux dicton ? Si on veut s'y référer de manière cohérente, on peut l'appliquer ainsi : on ne prend pas un hélicoptère pour aller acheter sa baguette au coin de la rue et, de la même manière, ce n'est pas à pied qu'on va tirer sa caravane pour partir au soleil. Vous avez certainement compris la démonstration. Cela dit, il n'est pas nécessaire d'utiliser le gros Pyradev pour assembler trois lignes de code ni Dams pour créer un programme de 30 Ko. Dans les deux cas, on se complique la vie pour pas grand-chose. Pyradev est très puissant, mais le fait qu'il travaille en fichiers séparés, dans un faux intégrateur, alourdit son utilisation. Si on veut débugger une toute ch'tiote routine, on s'arrache les cheveux. Alors que Dams, dans ce cas, devient un outil de rêve. Pour assembler plusieurs Ko de code, en revanche, il vaut mieux passer par le Monstre plutôt que par ce dernier. On sera ainsi sûr que le code source ne se fera pas bouffer par une quelconque fausse manipulation. Ce type de problème m'est arrivé, alors que mon code source était trop haut dans la mémoire, et le fait de charger un binaire avait radicalement pulvérisé des centaines de lignes de source.

SPEED LIMITED

Avec Dams, il faut faire attention. A partir de la gestion d'un source de 15 Ko, on peut avoir des surprises (je sens que je vais me faire traiter par les amoureux de ce produit). Il faut alors descendre le moniteur au maximum dans la mémoire. Le problème, dans ce cas, est de faire bien attention à la directive « M » qui décidera à partir de quel endroit les données d'assemblage ne devront plus monter vers le haut de la mémoire.

Lorsque vous travaillez avec des sources de taille conséquente sous Dams, je vous conseille fortement de le faire avec des copies de sauvegarde. Générez dans ces cas des noms de programmes avec une extension évolutive : Source.001, Source.002... Source.NNN. Avec cette méthode, vous serez certain de retrouver au moins une partie de votre code. Ce n'est malheureusement pas une solution mais simplement un palliatif. Si votre source gonfle trop, il vous sera toujours possible de le transformer pour le jeter sous un autre assembleur dont les capacités sont plus grandes (Pyradev, Devpac...). La manière dont Dams code ses programmes n'est pas bien sorcière. Il affecte en fait une valeur à chaque instruction, ce qui lui permet de garder une version compactée des programmes en mémoire. Si vous désirez en savoir plus, nous vous proposerons ces routines au sein de cette rubrique.

PASSE-MOI LE BEURRE

Le problème de la programmation en assembleur est que les programmes générés ne fonctionnent pas systématiquement et qu'il faut souvent s'y reprendre à au moins deux fois pour obtenir un truc qui fonctionne correctement. Lorsque le code est vraiment sain, pas de problème. On retourne sous Dams et on effectue les modifications. En revanche, il arrive que de grosses erreurs d'écriture détruisent jusqu'aux routines de Dams le Grand, lui-même en personne ! Dans ce cas, il faut se recogner le chargement de Dams, du source et enfin la modification du programme. Pour éviter au moins la première de ces étapes, nous allons vous offrir, au sein de ce numéro, un petit programme qui permet, aux possesseurs de 6128, de stocker l'assembleur dans les banques de mémoire parallèle. Ainsi, lorsqu'on plante le CPC sans avoir d'autres ressources que le « Control Shift Esc », il suffit d'un petit CALL pour que Dams revienne comme par magie au sein de la mémoire. Pour ceux qui possèdent un bouton Reset, il n'existe pas de cas dans lequel Dams ne reviendra pas en mémoire centrale. Ceci passe par une astuce que nous décrivons dès maintenant.

FAIS RISETTE A PAPA

Lorsqu'on réinitialise le CPC par la voie normale (CALL 0 ou en lançant la séquence de touches « Control Shift Esc »), toute la mémoire est nettoyée, sauf un groupe de Gaulois récalcitrants entourés de fortifications romaines ??? Pouf-pouf... Sauf la zone de mémoire qui s'étend de &BF00 à &BFFF pour tous les CPC et les banques mémoires pour les 6128. Cela veut donc dire que nous pouvons stocker dans la zone visible, juste sous la mémoire vidéo, quelques instructions. Un simple CALL en &BFXX nous permet dans ce cas de lancer une routine qu'il faut préalablement charger une fois à chaque extinction du CPC. Cette dernière sera protégée des attaques conventionnelles du système, ce qui nous permet de jouer tranquillement. En ce qui me concerne, je me suis amusé à loger la routine d'initialisation de Dams dans cette zone. Je peux ainsi déplacer Dams dans les banques et reprogrammer le pavé numérique du clavier pour qu'il fonctionne sous l'assembleur. Quoi de plus normal que de vous offrir cette routine...

```
;
     ORG  #BF00      ; Adresse d'assemblage
;
     JR   R2         ; Entrée secondaire
RUN
     DI              ; Entrée chargement
     LD   BC,#7FC1   ; Banque supérieure activée
     OUT  (C),C      ; sur le premier plan mémoire
     LD   HL,#1000   ; Adresse de chargement de Dams
     LD   DE,#C000   ; Adresse de stockage
     LD   BC,#4000   ; Taille à envoyer
     LDIR            ; Transfert
     LD   BC,#7FC0   ; Remise en forme
     OUT  (C),C      ; de la mémoire
     EI              ; On range tout
     RET             ; Ciao
R2                   ; Point d'entrée utilisateur
     OR   A          ; Paramètre dans DE ?
     JR   NZ,DEOK    ; Alors l'utiliser
     LD   DE,#1000   ; Par défaut DE = 1000
DEOK
     PUSH DE         ; Adresse à dépiler par RET final
     LD   BC,#7FC1   ; Commuter les banques
     OUT  (C),C      ; Comme pour le chargement
     LD   HL,#C000   ; De #C000
     LD   BC,#4000   ; Sur #4000 octets
     LDIR            ; Dams relogé en (DE)
     LD   BC,#7FC0   ; On range
     OUT  (C),C      ; C'est mieux

     LD   BC,0       ; Border 0
     CALL #BC38      ; Pour la beauté
     LD   BC,0       ; Fond noir
     XOR  A          ; C'est mieux
     CALL #BC32      ; Ink 0,0
     LD   BC,#1A1A   ; Ecriture en blanc
     LD   A,1        ; C'est propre
     CALL #BC32      ; Ink 1,27
;
     LD   A,#FF      ; IC nous commençons à enrichir Dams
     LD   (#B632),A  ; en redéfinissant le pavé numérique
     LD   DE,TOUCHE  ; DE pointe sur la table de redéfinition
     LD   C,12       ; Des 12 touches touches visées
REDEF
     LD   A,(DE)     ; Chargement de l'ID clavier
     LD   B,A        ; dans B
     INC  DE         ; avance
     LD   A,(DE)     ; chargement du code touche
     INC  DE         ; dans A
     CALL #BB27      ; Redéfinition
     DEC  C          ; et on boucle
     JR   NZ,REDEF   ; sur le clavier
     RET             ; Appel de Dams par CALL DE
;
TOUCHE
     DEFB 48,15,49,13,50,14,51,5
     DEFB 52,20,53,12,54,4,55,10,
     DEFB 56,11,57,3,13,6,35,7
;
```

COMMENTAIRE A TERRE

J'ai personnellement choisi de loger Dams en #1000, ceci pour me permettre une grande zone de travail en haut de mémoire. Malheureusement, cette pratique prend beaucoup de place au Basic, et il ne faut pas penser produire de gros programmes dans ce langage avec une telle configuration. Voici comment cela fonctionne. Le programme Basic de lancement de Dams charge l'assembleur en &1000 et fait un CALL en &BF02. Le programme est alors transféré dans la première banque mémoire supplémentaire. A chaque problème, plantage ou reset, le fait de forcer un CALL &BF00 branche sur le programme débutant en R2. Si un paramètre a été passé, il est dans DE. Dans ce cas, nous le gardons et il représente l'adresse de relogement de Dams. Dans le cas contraire, si aucun paramètre n'est précisé, nous réimplantons l'assembleur en #1000. Ceci fait, il ne nous reste qu'à hisser nos couleurs et à redéfinir les touches bien utiles du pavé numérique. Notez que nous utilisons le symbole « # » sur la touche « . » et que la touche Enter est remise en fonction.

ASTUCE POUR UN

Et un pour tous... Remarquez l'astuce qui nous permet de forcer automatiquement le lancement de Dams en empilant une adresse. Si nous réalisons le petit programme suivant :

```
LD    A,7
LD    HL,#BB5A
PUSH  HL
RET
```

cela revient à exécuter les lignes suivantes :

```
LD    A,7
JP    #BB5A
```

Suivez la pile et vous verrez que cela n'est pas magique. Il faut tout de même se méfier de telles programmations, car le type de bug engendré par ces lignes peut se révéler indétectable, même pour les yeux d'un programmeur averti. Je vous conseille donc de n'utiliser ces cabrioles que sur de très courtes routines, et si le besoin s'en fait vraiment sentir.

Voilà, la prochaine fois, nous ferons des petits programmes tout de même plus intéressants que ce type d'installation. M'enfin, cela fait partie des choses qu'il faut prendre en compte de manière à bien utiliser ses outils.

Aux bons travailleurs, salut...

Sined le Barbare à cas

SPRITES !

Salut bande de codeurs en délire ! Voici comment afficher un sprite selon diverses méthodes, en allant de la plus élémentaire et lente à la plus évoluée et rapide. Vous découvrirez aussi la diversité des modes d'affichage qu'offre le Z80.

Le listing se compose donc d'un programme principal et de différentes routines d'affichage de sprites que nous allons expliquer une par une. Qu'entendons-nous par « mode d'affichage » ? Eh bien ! c'est la façon dont va se combiner le motif du sprite avec le fond ou le décor affiché à l'écran : le sprite peut se superposer au fond, passer derrière, se mélanger avec, l'effacer, etc.

LE SPRITE EN MODE 1

Mais avant de voir les possibilités d'affichage, nous allons d'abord voir comment déterminer l'adresse-écran où il faut afficher le sprite en fonction des coordonnées *x* et *y* données. Nous allons étudier le cas d'un sprite en mode 1, car c'est un cas assez global. Un écran mode 1 normal fait 320 pixels de large, soit 80 octets. Un octet contient donc 4 pixels, et leur organisation binaire est si complexe qu'il est très compliqué d'afficher un sprite au pixel près : cela demanderait énormément de temps-machine pour faire tous les décalages, rotations et masquages nécessaires en temps réel. L'astuce consiste, comme d'habitude, à précalculer les positions des sprites au pixel près. Le même sprite est reproduit 4 fois en mémoire, à chaque fois décalé d'un pixel. Inconvénient : un sprite ainsi codé en mémoire prend 4 fois plus de place (2 fois plus de place s'il était en mode 0 puisqu'il n'y a que 2 pixels par octet, et 8 fois plus de place en mode 2 [8 pixels par octet]). Ainsi, le petit listing en Basic met en mémoire un petit sprite de 4 octets sur 13 lignes, décalé 4 fois, ainsi que son masque également décalé (nous en verrons l'utilité plus tard). En résumé, il faut donc décomposer l'abscisse *x* en nombre d'octets et en nombre de décalages de pixel par rapport au nombre de pixels contenus dans un octet (selon le mode).

Voilà pour l'abscisse, maintenant, occupons-nous de l'ordonnée (*y*). Comme vous le savez, l'organisation de la mémoire écran n'est pas linéaire : c'est-à-dire que la 2ᵉ ligne visible à l'écran ne se trouve pas en mémoire juste après la première. Bien qu'il soit relativement facile de descendre d'une ligne à une autre, il est plus difficile de se positionner directement à la ligne voulue par un algorithme, d'où la nécessité encore une fois de précalculer une table des ordonnées contenant l'adresse écran de chaque ligne, les unes en dessous des autres, permettant d'obtenir automatiquement et directement l'adresse de la ligne voulue ! On récupère l'adresse de la ligne voulue en prenant l'ordonnée et en la multipliant par 2 (puisqu'une adresse tient sur 2 octets) et on additionne à cette valeur l'adresse où est stockée la table ; on pointe ainsi sur l'adresse voulue qu'on n'a plus qu'à récupérer. Cette opération peut se faire de plusieurs façons ; dans le listing 1, c'est le registre HL qui pointe sur la table des adresses des lignes écran tandis que dans le listing 2, les adresses sont récupérées grâce à la pile (ce qui peut s'avérer plus rapide, à condition que la pile ne soit pas utilisée en même temps). Voilà donc obtenues l'adresse source (l'adresse du sprite décalé correspondant à l'abscisse) et l'adresse destination (l'adresse écran ou va être affiché le sprite). On peut donc maintenant passer à l'affichage proprement dit.

L'AFFICHAGE

Le listing 1 regroupe plusieurs façons conventionnelles d'afficher un sprite, nous allons donc les commenter une par une, des plus simples aux moins évidentes avec leurs points forts et leurs points faibles :

- n°1 : affichage par LDIR. Vraiment la méthode la plus élémentaire pour afficher un sprite ! Avantage : le code ne prend pas beaucoup de place en mémoire. Inconvénient : un LDIR demande l'équivalent de 5 NOPS en temps-machine, ce qui est quand même pas mal, d'autant plus que le sprite est affiché à l'écran en effaçant le fond sur tout le rectangle qui définit sa surface ;

- n°2 : affichage par LDI. Le principe est à peu près le même que celui de la routine n°1, à ceci près qu'on remplace le LDIR par autant de LDI que le LDIR doit être effectué (autrement dit, on met autant de LDI qu'il y a d'octets qui composent le sprite en largeur). Avantage : sensiblement plus rapide que LDIR (un LDI prend 4 NOPS en temps-machine), surtout si le sprite est large. Inconvénient : la routine d'affichage prend d'autant plus de place que le sprite est large, et le fond à l'endroit où le sprite s'affiche disparaît là encore ;

- n°3 : affichage avec test d'octet. Ce système permet de donner au sprite un semblant de masquage. On teste à l'aide d'un OR si l'octet du sprite est nul. S'il n'est pas nul, on l'affiche, sinon, on passe au suivant. Avantage : le fond derrière le sprite est ainsi partiellement sauvé, et la routine occupe peu de mémoire. Inconvénient : la routine est lente et le résultat à l'écran n'est pas toujours parfait, car le masquage se fait a l'octet et non au pixel (remarque : le résultat reste fort convenable en mode 0).

- n°4 : affichage en « mélangeant » sprite et fond. Ici, on fait directement un OR entre le sprite et le fond, et on affiche le résultat. Ce genre de routine est avantageuse lors d'un écran en « dual-playfield » : grâce à un jeu d'encre, on peut afficher des sprites masqués au pixel par un simple OR, mais ces sprites sont moins colorés que des sprites normaux. Dans le cas d'un dual playfield (*cf.* les sprites de la démo du dernier numéro d'*Amstrad Cent Pour Cent*), il suffit d'utiliser exactement la même routine avec un XOR à la place du OR pour réafficher le fond. S'il n'y a pas de dual-playfield, le résultat à l'écran peut parfois être bizarre (superpositions de couleurs...), mais cette routine a l'avantage d'être suffisamment rapide et de ne pas prendre trop de mémoire.

- n°5 : affichage par masquage. La routine qui offre le meilleur résultat visuel, mais aussi la plus compliquée ! Elle nécessite, en plus des données du sprite, un masque qui détermine quels pixels du fond doivent disparaître (cela permet, par exemple, de faire un petit contour noir autour des sprites, comme dans le jeu Knight-Time). On fait un AND entre le masque et le fond, puis on fait un OR

LOGON

entre le résultat et la donnée de l'octet du sprite, et enfin, on affiche. Avantage : finesse de l'affichage. Inconvénients : le masque prend autant de place que le sprite et l'affichage est lent.

- n°6 : affichage à la pile. un affichage qui trouve son utilité, surtout pour déplacer de gros sprites sur un fond uni. En effet, cette technique d'affichage prend les octets 2 par 2 (d'où la nécessité d'avoir un sprite composé d'un nombre pair d'octets en largeur), ce qui permet d'être plus rapide, mais le fond derrière le sprite est détruit.

Voilà pour ce qui est de l'affichage des sprites. Il est possible, par exemple dans le cadre d'une animation, que vous souhaitiez que le fond soit restitué. Dans ce cas, il suffit de le sauver à l'aide de LDIRs ou de LDIs dans un buffer de la taille du sprite, puis que vous affichiez le sprite, et ensuite que vous réaffichiez le fond précédemment sauvegardé grâce à des LDIs, LDIRs, PUSH-POP, etc.

BAS LES MASQUES

On a donc vu que les routines les plus intéressantes, notamment celles par masquage, sont souvent les plus lentes. Pour avoir un masquage rapide, nous allons faire un générateur de codes, qui produit une routine telle que le sprite est incrusté dans cette routine. C'est ce que fait la routine « gener » du listing 2. Cette routine crée donc 4 routines de sprites différentes (une pour chaque décalage de pixel). Les adresses de ces 4 routines sont contenues dans une table qui sert lors du traitement de l'abscisse, à savoir sur quelle routine d'affichage il faudra sauter plus tard. Comme les masquages entre le fond et le sprite sont immédiats, l'affichage est extrêmement plus rapide de cette façon. En revanche, la mémoire demandée est assez considérable puisqu'il n'y a pas de boucle et puisqu'il faut environ 8 octets de code autogénéré pour afficher un octet du sprite. Néanmoins, on peut se permettre cette perte d'espace mémoire pour des petits sprites, et à plus forte raison si on fait une démo.

Le listing 2 vous explique le code autogénéré pour l'affichage par masquage, mais de toutes façons, il est également possible de faire du code autogénéré avec les autres types d'affichage. Par exemple, pour l'affichage avec test d'octet, il suffit de vérifier si l'octet est vide ou non lors de la génération du code ; le code résultant se trouve grandement optimisé !

Voilà, nous avons vu l'essentiel concernant l'affichage des sprites, il ne me reste plus qu'à vous souhaiter de bien assimiler cette rubrique et à espérer que j'en aurai bientôt fini avec cette ?#*!$@ de grippe !

A bientôt ! Pict

```
; Exemple de routines d'affichage de sprites
; (c) Pict/Logon System 1993 pour A100%
; Assemble avec DAMS charge en #4000
;
;          Coordonnees du sprite
x           EQU  40
y           EQU  2
;
;          Dimensions du sprite
haut        EQU  13
larg        EQU  4
size        EQU  larg*haut
mask        EQU  4*size
;
;          Adresse de la pile
;          du programme
;
stack       ORG  #2000
;
;          Adresse des donnees
;          du sprite
;
sprite      EQU  #1000
tabspr      DEFW size*0+sprite
            DEFW size*2+sprite
            DEFW size*1+sprite
            DEFW size*3+sprite
;
;          Largeur de l'ecran
;          en word.
r1          EQU  40
;
;          Routine de chargement du sprite
;
noml        DEFM sprite  .BIN
load        LD   B,#0C
            PUSH DE
            CALL #BC77
            POP  HL
            CALL #BC83
            JP   #BC7A
;
;          Debut du programme
run
            ENT  $
            LD   hl,noml
            LD   de,sprite
            CALL load
;
;          Passage en mode 1 et affichage
;          d'un fond quelconque
;
go          LD   a,1
            CALL #bc0e
            LD   hl,fond
affond      LD   a,(hl)
            INC  hl
            OR   a
            JP   z,finaff
            PUSH hl
            CALL #bb5a
            POP  hl
            JP   affond
finaff
;
;          Bloquage du FirmWare pour
;          avoir la jouissance de tous
;          les registres du Z-80 qu'on
;          sauvegarde.
;
            DI
            LD   hl,(#38)
            LD   (sys1+1),hl
            LD   hl,#C9FB
            LD   (#38),hl
            LD   (stasys1+1),sp
            LD   sp,stack
            EXX
            EX   af,af
            PUSH hl
            PUSH de
            PUSH bc
            PUSH af
;
;          Initialisation des
;          couleurs du sprite
;
            LD   bc,#7f00
            LD   de,#5458
            LD   hl,#4d4b
            OUT  (c),c
            OUT  (c),d
            INC  c
            OUT  (c),c
            OUT  (c),e
            INC  c
            OUT  (c),c
            OUT  (c),h
            INC  c
            OUT  (c),c
            OUT  (c),l
;
;          Creation de la table
;          des ordonnees
;          (hauteur normale=200 lignes)
;          (adresse normal=#c000
;
            LD   hl,#c000
            LD   de,r1*2+#c000
            LD   bc,#800
            LD   ix,taby
            LD   a,200
creey       LD   (ix+1),h

            LD   (ix+0),l
            INC  ix
            INC  ix
            ADD  hl,bc
            JP   nc,nocarry
            ADD  hl,de
nocarry     DEC  a
            JP   nz,creey
;
;          Appel du listing d'Affichage
;          (Listing1,Listing2)
;
            CALL listing2
;
;          Attente d'appui sur Espace
;
key         LD   bc,#f782
            OUT  (c),c
            LD   bc,#f40e
            OUT  (c),c
            LD   bc,#f6c0
            OUT  (c),c
            XOR  a
```

```
        XOR  a
        OUT  (c),a
        LD   bc,#f792
        OUT  (c),c
        LD   de,#f4f6
        LD   c,#45
        LD   b,e
        OUT  (c),c
        LD   b,d
        IN   d,(c)
        LD   bc,#f782
        OUT  (c),c
        DEC  c
        OUT  (c),a
        RL   d
        JP   c,key
;
;       mets le pen 1 blanc.
        LD   bc,#7f01
        LD   a,#4b
        OUT  (c),c
        OUT  (c),a
;
;       Retablissement du Firmware
;
firm
        DI
        POP  af
        POP  bc
        POP  de
        POP  hl
        EXX
        EX   af,af

stasysl LD   sp,0
sysl    LD   hl,0
        LD   (#38),hl
        EI
        RET
;
fond    DEFM LOGON SYSTEM
        DEFB 10,13
        DEFM AMSTRAD 100%
        DEFB 0
taby    DEFS 2*200,0
;
;       Programme principal
;       On transforme les
        coordonnees x et y
        en adresse ecran.
;
;
listing1
        On s'occupe d'abord de
;       l'abscisse
;
        LD   bc,x
        XOR  a
        LD   d,a
        SRL  b
        RR   c
        RLA
        SRL  b
        RR   c
        RLA
        ADD  a,a
        les 2 derniers bits de x
;       determinent le sprite a
;       afficher parmi les sprites
;       decales au pixel dans la
;       memoire
;
        LD   e,a
        LD   hl,tabspr
        ADD  hl,de
        LD   e,(hl)
        INC  hl
        LD   d,(hl)
        LD   a,c
;
;       L'ordonnee permet de pointer
;       dans une table sur l'adresse
;       de la ligne ecran correspondan
        LD   hl,y
        ADD  hl,hl
        LD   bc,taby
        ADD  hl,bc
        LD   c,(hl)
        INC  hl
        LD   b,(hl)
;
;       on additionne a cette adresse

;       l'abscisse en octets
        LD   h,0
        LD   l,a
        ADD  hl,bc
;       on a l'adresse finale dans HL
        LD   a,haut
;       ...qu'on transfere dans DE
        EX   de,hl
;
;       ROUTINES D'AFFICHAGE DU SPRITE
;
;       (jp loop1_1,loop1_2,...,loop1_6)
;
routine JP   loop1_5
;
;       1ERE ROUTINE
;       AFFICHAGE LE PLUS SIMPLE
;       LDIR SEULEMENT
;
loop1_1 LD   bc,larg
        LDIR
        LD   bc,#800-larg
        EX   de,hl
        ADD  hl,bc
        JP   nc,nocarry1
        LD   bc,r1*2+#c000
        ADD  hl,bc
nocarry1 EX  de,hl
        DEC  a
        JP   nz,loop1_1
        RET
;
;       2EME ROUTINE:A PEU PRES INDENTIQUE
;       A LA PREMIERE MAIS ON REMPLACE LDIR
;       PAR AUTANT DE LDI QUE LE SPRITE EST
;       LARGE EN OCTETS.
;
loop1_2 LDI
        LDI
        LDI
        LDI
        LD   bc,#800-larg
        EX   de,hl
        ADD  hl,bc
        JP   nc,nocar1_2
        LD   bc,r1*2+#c000
        ADD  hl,bc
nocar1_2 EX  de,hl
        DEC  a
        JP   nz,loop1_2
        RET
;
;       3EME ROUTINE:ON TESTE CHAQUE OCTET
;       DU SPRITE A AFFICHER:S'IL EST NUL,
;       ON NE L'AFFICHE PAS

;
loop1_3 EX   de,hl
loop1_31 PUSH af
        LD   b,larg
loop1_32 LD  a,(de)
        INC  de
        OR   a
        JP   z,noaff1_3
        LD   (hl),a
noaff1_3 INC hl
        DJNZ loop1_32
        LD   bc,#800-larg
        ADD  hl,bc
        JP   nc,nocar1_3
        LD   bc,#c050
        ADD  hl,bc
nocar1_3 POP af
        DEC  a
        JP   nz,loop1_31
        RET
;
;       4EME ROUTINE:LE SPRITE ET LE
;       FOND SONT "MELANGES" AVEC UN
;       OR;LE RESULTAT N'EST PAS
;       TOUJOURS GRACIEUX,MAIS LA
;       RAPIDITE D'AFFICHAGE EST
;       CONVENABLE...
;       ON AURAIT PU UTILISER XOR A
;       LA PLACE DE XOR CE QUI
;       D'EMPLOYER LA MEME ROUTINE
;       POUR L'AFFICHAGE ET L'EFFACAGE
;       MAIS LE RESULTAT A L'ECRAN EST
;       SOUVENT TRES LAID...
;
loop1_4 EX   de,hl
loop1_41 PUSH af
        LD   b,larg
loop1_42 LD  a,(de)
        OR   (hl)
        LD   (hl),a
        INC  hl
        INC  de
        DJNZ loop1_42
        LD   bc,#800-larg
        ADD  hl,bc
        JP   nc,nocar1_4
        LD   bc,r1*2+#c000
        ADD  hl,bc
nocar1_4 POP af
        DEC  a
        JP   nz,loop1_41
        RET
;
;       5EME ROUTINE:AFFICHAGE MASQUE:
;       LE PLUS EFFICACE,MAIS AUSSI LE

;       PLUS LENT!...A MOINS D'UTILISER
;       DU CODE AUTOGENERE!(VOIR LISTING 2)
;
loop1_5
;       on recupere le masque
;       correspondant au sprite decale
        LD   bc,mask
        PUSH hl
        ADD  hl,bc
        LD   b,h
        LD   c,l
        POP  hl
        EX   de,hl
        EXX
        DI
        LD   (sp1_5+1),sp
        LD   sp,#800-larg
loop1_51 EX  af,af
        LD   b,4
loop1_52 EXX
;
;       DE=sprite
;       BC=masque
;       HL=adresse ecran
;
        LD   a,(bc)
        AND  (hl)
        EX   de,hl
        OR   (hl)
        EX   de,hl
        LD   (hl),a
        INC  hl
        INC  de
        INC  bc
        EXX
        DJNZ loop1_52
        EXX
        ADD  hl,sp
        JP   nc,nocar1_5
        LD   sp,r1*2+#c000
        ADD  hl,sp
        LD   sp,#800-larg
nocar1_5
        EXX
        EX   af,af
        DEC  a
        JP   nz,loop1_51
sp1_5   LD   sp,0
        EI
        RET
;
;       6EME ROUTINE:
;       AFFICHAGE A LA PILE:RAPIDE MAIS
;       LE FOND EST DETRUIT ET LE SPRITE
;       DOIT ETRE COMPOSE D'UN NOMBRE

;       PAIR D'OCTETS EN LARGEUR
loop1_6 DI
        LD   (sp1_6+1),sp
        LD   sp,hl
        EX   de,hl
        LD   b,a
loop1_61 LD  c,larg/2
loop1_62
        POP  de
        LD   (hl),e
        INC  hl
        LD   (hl),d
        INC  hl
        DEC  c
        JP   nz,loop1_62
        LD   de,#800-larg
        ADD  hl,de
        JP   nc,nocar1_6
        LD   de,#c050
        ADD  hl,de
```

```
nocar1_6  DJNZ loop1_61
sp1_6     LD   sp,0
          EI
          RET
;
;         Listing2:Sprite Autogenere:
;         les donnees du sprite sont
;         incrutees dans la routine.
;
listing2
          CALL gener
          CALL affiche
          RET
;
;         routine d'affichage en
;         code genere:cette partie
;         determine laquelle des 4
;         routines doit etre
;         appelee.
;         on se sertt de la pile
;         pour recuperer l'adresse
;         de la ligne de l'ecran
;         et celle de la routine
;         d'affichage dont on a
;         besoin
;
affiche
          DI
          LD   (sp2+1),sp
;
;         on recupere d'abord
;         l'adresse de la
;         routine d'affichage
;         tout en transformant
;         l'abscisse.

          LD   de,x
          LD   a,e
          SRL  d
          RR   e
          SRL  d
          RR   e
          LD   sp,tabadsp
          AND  3
          ADD  a,a
          LD   h,d
          LD   l,a
          ADD  hl,sp
          LD   sp,hl
          POP  ix
;
;         on recupere maintenant
;         l'adresse de la ligne
;         de l'ecran.
          LD   sp,taby
          LD   hl,y
          ADD  hl,hl
          ADD  hl,sp
          LD   sp,hl
          POP  hl
          ADD  hl,de
          LD   bc,#800-larg+1
          LD   de,r1*2+#c000
sp2       LD   sp,0
          EI
;
;         saut a la routine d'affichage
;         selon l'abscisse
          JP   (ix)
;
;         Programme de generation
;         des 4 routines de sprites.
;
gener     LD   iy,sprout
          LD   hl,sprite
          LD   de,sprite+mask
          LD   ix,tabadsp
          LD   a,4
codebyte
          PUSH af
          PUSH hl
          PUSH iy
          POP  hl
          LD   (ix+0),l
          LD   (ix+1),h
          POP  hl
          INC  ix
          INC  ix
          PUSH ix
          CALL calc
          POP  ix

          POP  af
          DEC  a
          JP   nz,codebyte
          RET
;
calc      LD   b,haut
pokeline  PUSH bc
          LD   b,larg
poke      LD   a,(de)
;
;         on teste d'abord si
;         le masque est plein
;
          CP   #ff
;
;
;         si oui,pas la peine
;         d'afficher l'octet,
;         on passe au suivant.
;
          JP   z,nxtbyte
;
;
;         on teste ensuite si
;         le masque est nul:
;         dans ce cas,pas besoin
;         de masquer,on affiche
;         la donnee de l'octet du
;         sprite directement
;
          OR   A
          JP   nz,maskbyte
          LD   a,(hl)
          LD   (iy+0),#36
          INC  iy
          LD   (iy+0),a
          INC  iy
nxtbyte   LD   (iy+0),#23
          INC  iy
          JP   coded
maskbyte
          PUSH bc
;
;         On recopie le "bout"
;         de code qui affiche et
;         masque un octet du sprite
;
pkr       LD   a,(ix+0)
          LD   (iy+0),a
          INC  ix
          INC  iy
          DJNZ pkr
          POP  bc
;
;         on incruste dans le code
;         le masque et la donnee
;         de l'octet du sprite.
          LD   a,(de)
          LD   (iy-5),a
          LD   a,(hl)
          LD   (iy-3),a
coded
          INC  de
          INC  hl
          DJNZ poke
;
;         on passe a la ligne
;         le "INC L" inutile
;         en decrementant IY.
          DEC  iy
          LD   b,4
;
;         on recopie le "bout"
;         de code qui permet de
;         descendre d'une ligne.
          LD   ix,nxtline
pknx      LD   a,(ix+0)
          LD   (iy+0),a
          INC  ix
          INC  iy
          DJNZ pknx
          POP  bc
          DJNZ pokeline
;
;         fin de la routine
;         d'affichage.
;         on n'a pas besoin
;         de redescendre d'une
;         ligne donc on elimine
;         le morceau de code
;         precedent en reculant le
;         pointeur du code genere
;         pour reecrire par dessus.
;
          LD   BC,-4
          ADD  IY,BC
          LD   (IY+0),#c9
          INC  iy
          RET
;
;         "morceau" de programme
;         necessaire pour masquer
;         et afficher un octet.
;
oper      ; bytes
          LD   a,(hl)
          AND  0
          OR   0


          LD   (hl),a
          INC  l
;
;         "morceau" de programme
;         necessaire pour desecndre
;         d'une ligne.
nxtline   ; bytes
          ADD  hl,bc
          JR   nc,noaddde
          ADD  hl,de
noad  ;
;         "morceau" de programme
;         necessaire pour desecndre
;         d'une ligne.
nxtline   ; bytes
          ADD  hl,bc
          JR   nc,noaddde
          ADD  hl,de
noaddde
end
;
;         table cree par le
;         generateur qui
;         contient les adresses
;         des 4 routines generees.
tabadsp   DEFS 4*2,0
;
;         les routines sont generees
;         dans l'espace de memoire
;         debutant ici-meme.
sprout


10 MEMORY &3FFF
20 FOR a=&4000 TO &4000+4*8*13
30 READ a$:POKE a,VAL("&"+a$)
40 NEXT
50 SAVE"sprite.bin",b,&4000,8*13*4
60 DATA 00,70,00,00,10,80,C0,00,20,00,24
,00,40,00,5E,00,40,00,14,00,80,00,00,80,
C4,00,00,80,84,00,00,80,42,00,10,00,43,0
0,12,00,21,6F,2C,00,10,87,C0,00,00,70,00
,00,00
70 DATA 30,80,00,00,C0,60,00,10,00,12,00
,20,00,27,80,20,00,02,80,40,00,00,40,62,
00,00,40,42,00,00,40,21,00,00,80,21,08,0
1,80,10,3F,1E,00,00,C3,68,00,00,30,80,00
,00,10
80 DATA C0,00,00,60,30,00,00,80,01,80,10
,00,13,48,10,00,01,40,20,00,00,20,31,00,
00,20,21,00,00,20,10,08,00,40,10,0C,00,4
8,00,97,8F,80,00,61,3C,00,00,10,C0,00,00
,00,E0
90 DATA 00,00,30,10,80,00,40,00,48,00,80
,01,AC,00,80,00,28,10,00,00,10,10,88,00,
10,10,08,00,10,00,84,00,20,00,86,00,24,0
0,43,CF,48,00,30,1E,80,00,00,E0,00,FF,88
,FF,FF
100 DATA EE,00,33,FF,CC,00,11,FF,88,00,0
0,FF,88,00,00,FF,00,00,00,77,00,00,00,77
,00,00,00,77,88,00,00,FF,88,00,00,FF,CC,
00,11,FF,EE,00,33,FF,FF,88,FF,FF,FF,CC,7
7,FF,FF
110 DATA 00,11,FF,EE,00,00,FF,CC,00,00,7
7,CC,00,00,77,88,00,00,33,88,00,00,33,88
,00,00,33,CC,00,00,77,CC,00,00,77,EE,00,
00,FF,FF,00,11,FF,FF,CC,77,FF,FF,EE,33,F
F,FF,88
120 DATA 00,FF,FF,00,00,77,EE,00,00,33,E
E,00,00,33,CC,00,00,11,CC,00,00,11,CC,00
,00,11,EE,00,00,33,EE,00,00,33,FF,00,00,
77,FF,88,00,FF,FF,EE,33,FF,FF,FF,11,FF,F
F,CC,00
130 DATA 77,FF,88,00,33,FF,00,00,11,FF,0
0,00,11,EE,00,00,00,EE,00,00,00,EE,00,00
,00,FF,00,00,11,FF,00,00,11,FF,88,00,33,
FF,CC,00,77,FF,FF,11,FF,00,00,00,00,00,0
0,00,00
```

LISTING

NOS LECTEURS SONT ROIS

Les plus anciens se souviendront avec nostalgie des prouesses du Grand Barbare dans la rubrique « Les sept nains ». Voici pour vous le « best of » des plus petits programmes sur CPC, à consommer avec modération. Notez avant de vous régaler que dans le prochain numéro d'*Amstrad Cent Pour Cent,* nous vous proposerons un super jeu d'arcade réalisé de main de maître par un certain Moktar (non, non, c'est pas lui). En attendant, encodez ce véritable collector.

Pour une fois, vous ne pourrez pas vous plaindre de passer des nuits blanches à saisir notre listing. Pour cause ! L'encodage de chaque bloc devrait vous prendre moins de cinq minutes. Notez que les programmes sont indépendants les uns des autres et que vous devez les sauvegarder un par un sur le support de votre choix (disquette / cassette).

Voici un rapide descriptif de nos progs. Dessiner une multitude d'ellipses de formats différents, faire des gri-gris dans le border histoire d'épater la copine, animer une hélice (cette routine peut être ajoutée au premier prog), faire chanter l'oiseau qui est dans votre CPC, grâce aux trames et à l'assembleur, donner un look plein de couleurs à votre machine, une musique by Poum, donc à ne pas manquer, une routine vous démontrant la technique à employer pour créer une animation spatiale en utilisant la palette des couleurs, réveiller l'oiseau qui dort dans votre CPC chéri, connaître à tout moment sous forme de variable les encres utilisées (INK), encore une zic et un truc sympa pour générer des nombres aléatoires, suivi d'un prog pour afficher vos pages écran.

Pour le reste vous aurez la surprise en temps voulu, mais n'oubliez sous aucun prétexte les quelques lignes qui gèrent des paysages désertiques.

Ne nous remerciez pas, c'est normal.

CE PROGRAMME DESSINE DES ELIPSES DE MOINS EN MOINS ARRONDIES

```
10 ' Elipse
20 MODE 0:INK 0,0:BORDER 0
30 DEFINT X-Y:X=320:Y=200
40 RAD:FOR A=0.2 TO PI*2 STEP 0.2
50 MOVE 320,200
60 FOR R=0 TO 200 STEP A
70 DRAW R*COS(R)+X,R*SIN(R)+Y,Z AND 15
80 Z=Z+1
90 NEXT R
100 WHILE INKEY$="":WEND:CLS
110 NEXT A
```

CE PROGRAMME CREE DES EFFETS DANS LE BORDER

```
10 '* BORDER par robin STEFANETTI
20 MEMORY &9FFF:FOR I=&A000 TO &A012
30 READ A$:A=VAL("&"+A$):POKE I,A
40 NEXT
50 DATA 06,7F,0E,10,ED,49,0E,40,ED,49,0E
,10,ED,49,0E,54,ED,49,C9
60 '*** Exemple ***
70 FOR i=1 TO 5000:CALL &A000:NEXT:END
```

CE PROGRAMME DESSINE UNE HELICE ET LA FAIT ANIMER PAR CYCLE DE COULEURS

```
20 MODE 0:INK 0,0:BORDER 0:RAD:MOVE 320,
200:FOR i=0 TO 100 STEP 0.1:DRAW i*COS(i
)+320,i*SIN(i)+200,c AND 15:c=c+1:NEXT
30 WHILE INKEY$="":FOR y=1 TO 15:INK y,2
0:CALL &BD19:INK y,0:NEXT:WEND:INK 1,26
```

VOUS M'AVEZ PAS CRU
VOUS M'AUREZ CUIT

```
30 RANDOMIZE TIME
40 FOR n=1 TO 1000
50 SOUND 1,RND*n,1:SOUND 2,RND*n,1:SOUND
 4,RND*n,1
60 NEXT
```

ROUTINE DE TRAME PAR REGIS DE MBM
UNE BONNE TONNE DE COULEUR

```
100 '
110 ADR= 16384:FOR I=0 TO 34
120 FOR J=1 TO 8:READ A$
130 A=VAL("&"+A$)
140 B=(B+I+A*J) AND 255
150 POKE ADR+I*8+J-1,A:NEXT J
160 READ B$:IF B=VAL("&"+B$) THEN 180
170 PRINT "ERREUR EN ";1000+I*10:STOP
180 NEXT I:SAVE"regis",B,ADR, 272
190 CALL ADR
1000 DATA F3,01,9E,7F,ED,49,01,10,A9
1010 DATA 7F,3E,54,ED,49,ED,79,21,AE
1020 DATA 00,C0,11,01,C0,01,FE,3F,25
1030 DATA 36,AA,ED,B0,21,7A,40,06,BF
1040 DATA 02,C5,06,C8,CB,F6,23,10,CD
1050 DATA FB,C1,21,C5,40,10,F2,06,57
1060 DATA F5,ED,78,1F,D2,2F,40,06,5E
1070 DATA FF,00,00,00,00,00,00,95
1080 DATA 00,00,00,00,00,00,10,F1,CD
1090 DATA 06,36,10,FE,00,00,00,D9,77
1100 DATA 21,C5,40,1E,01,06,7F,48,8C
1110 DATA D9,21,7A,40,3E,4B,1E,00,37
1120 DATA 0E,7F,41,ED,59,ED,A3,D9,A2
1130 DATA ED,59,ED,A3,41,D9,06,09,C9
1140 DATA 10,FE,00,3D,C2,62,40,C3,27
1150 DATA 2F,40,14,14,16,19,12,19,24
1160 DATA 19,03,03,0B,0B,0B,0B,0B,16
1170 DATA 0B,03,03,19,19,12,19,16,64
1180 DATA 14,14,14,18,18,0D,0D,0F,65
1190 DATA 0F,0B,0B,0F,0F,0D,0D,18,33
1200 DATA 18,14,14,1C,0C,0C,0C,0C,F7
1210 DATA 0C,0E,0E,0A,0A,0A,03,0B,F4
1220 DATA 0B,0B,03,0A,0A,0A,0E,0E,36
1230 DATA 0C,0C,0C,0C,0C,14,14,14,46
1240 DATA 14,14,14,14,14,16,12,16,E4
1250 DATA 16,12,12,19,12,19,12,19,B6
1260 DATA 0B,0B,19,12,19,12,19,12,62
1270 DATA 12,16,16,12,16,14,14,18,34
1280 DATA 18,0D,0D,0F,0F,0B,0B,0F,FB
1290 DATA 0F,0D,0D,18,18,14,14,14,AF
1300 DATA 1C,0C,0E,0A,0E,0A,0A,03,05
1310 DATA 0B,0B,0B,0B,0B,0B,0B,03,49
1320 DATA 0A,0A,0E,0A,0E,0C,1C,14,AB
1330 DATA 14,14,14,14,14,14,14,14,83
1340 DATA 00,00,00,00,00,00,00,00,93
```

UNE ZIZIC POUR VOS PETITES OREILLES

```
30 INK 0,0:INK 1,26:PAPER 0:PEN 1:BORDER
 0:MODE 1:LOCATE 11,12:PRINT"ARRETER PAR
 ESPACE"
40 GOSUB 510
50 '
60 DATA 15,13,9,15,13,16,9,13,23,16,13,1
5
70 DATA 16,13,15,15,13,14,16,13,15,11,9,
25
80 RESTORE 60:FOR I=1 TO 8:READ IN1(I),I
N2(I),IN3(I):NEXT
90 '
100 '
110 '
120 DATA 851,638,536,426,319,268,213,268
,319,426,536,638
130 DATA 851,716,568,426,358,284,213,284
,358,426,568,716
140 DATA 956,716,568,478,358,284,239,284
,358,478,568,716
150 DATA 956,758,638,478,379,319,239,319
,379,478,638,758
160 DIM BAS(96):RESTORE 120:FOR J=0 TO 1
:FOR I=1 TO 24:READ A
170 BAS(I+J*48)=A:BAS(I+J*48+24)=A:NEXT
I,J
180 BAS=0:TEMPS=32:GOSUB 480:EVERY 16 GO
SUB 480
190 ENV 1,1,10,1,1,0,15,16,-1,6:ENV 2,4,
1,2,4,-1,2
200 ENT 1,5,3,1,5,-3,1:ENT -2,2,2,2,4,-2
,2,2,2,2
210 EVERY 6,1 GOSUB 520
220 '
230 DATA 106,10,100,2,106,12,106,10,119,
2,106,12
240 DATA 119,10,106,2,95,10,106,2,119,10
,106,2,119,10,159,2
250 DATA 134,10,119,2,142,10,134,1,142,1
,159,10,134,2,142,10,179,2
260 DATA 119,10,106,2,95,10,89,1,95,1,11
9,10,106,2,119,12
270 RESTORE 230:FOR I=1 TO 31:READ NOTE,
MULTI
280 SOUND 4,NOTE,-2*MULTI,7,2,1:NEXT
290 '
300 ENV 3,12,1,10,2,-1,20,24,0,1,10,-1,2
0
310 FOR I=0 TO 3:SOUND 4,0,-1,1,3,,31-I*
3:NEXT I
320 '
330 ENV 4,12,1,16,12,-1,16
340 SOUND 4,119,-1,2,4,2:SOUND 4,0,-1,0,
3,,31
350 SOUND 4,60,-1,2,4,2:SOUND 4,0,-1,0,3
,,20
360 '
370 DATA 106,119,106,159,100,89,106,134,
159
380 DATA 142,213,134,119,142,106,119,134
```

```
150 NEXT
160 SOUND EN(1),NOTE(1),DUR(1),6,1,E(1)
170 SOUND EN(2),NOTE(2),DUR(2),6,1,E(2)
180 SOUND EN(3),NOTE(3),DUR(3),6,1,E(3)
190 FOR P=1 TO INT(RND*150)+100:NEXT
200 NEXT
210 FOR P=1 TO INT(RND*1000)+500:NEXT
220 GOTO 80
```

TRES PRATIQUE POUR CONNAITRE LA VALEUR DES 16 ENCRES (INK) A TOUT MOMENT

```
20 DIM enc(16),cod(32)
30 FOR i=0 TO 31:READ a$:cod(VAL("&"+a$)
)=i:NEXT
40 DATA 14,04,15,1C,18,1D,0C,05
50 DATA 0D,16,06,17,1E,00,1F,0E
60 DATA 07,0F,12,02,13,1A,19,1B
70 DATA 0A,03,0B,01,08,09,10,11
80 A=&B7D5:IF PEEK(6)=128 THEN A=&B1EB
90 MODE 1:LOCATE 1,1:FOR i=0 TO 15:enc(i
)=cod(PEEK(a+i)):PRINT"INK"i","enc(i),:N
EXT
```

ENCORE UNE ZIC (LA SECONDE)

```
100 ENV 1,1,15,1,5,-1,4,10,-1,10:ENT -1,
1,20,1
110 ENV 2,1,13,1,5,-1,2:ENT 2,2,10,1,2,-
10,1
120 ENV 3,1,12,1,9,-1,1:ENT -3,1,1,2
130 ENT 4,3,-12,1,3,12,1:ENV 4,1,12,1,12
,-1,4
140 ENV 5,3,5,1,15,-1,6
150 ENV 6,15,1,1,15,-1,30:ENT -6,4,50,1,
4,-100,1,4,50,1
160 ENV 7,15,1,20,10,-1,19:ENT -7,2,4,2,
2,-8,2,2,4,2
170 DATA 426,536,478,851
180 FOR I=1 TO 4:READ BASSE(I):NEXT I
190 '
200 SOUND 1,852,-1,0,7,7
210 SOUND 2,851*2,-1,0,7,7
220 SOUND 4,851,-1,0,7,7
230 '
240 DATA 190,179,159,190,190,179,159,179
250 DATA 190,179,159,106,142,179,213,213
260 FOR K=0 TO 2:FOR I=1 TO 4:IF I<>4 TH
EN RESTORE 240
270 FOR P=1 TO 4:SOUND 4,BASSE(I)/(64/(P
)),40,7,,3:NEXT P

280 FOR J=1 TO 8:READ NO:DIV=1:IF J<4 TH
EN DIV=K
290 IF K=0 THEN DIV=1:CAN=3:GOTO 320 ELS
E CAN=1
300 BAT=0:BB=BB XOR 1:IF BB=1 THEN BAT=1
+BA*30:BA=BA XOR 1
310 SOUND 2,BASSE(I),-2,0,3,,BAT
320 SOUND CAN,213,10,0,4,4:SOUND CAN,NO/
DIV,10,0,4,4
330 IF INKEY(47)=0 THEN CALL &BCA7:END
340 NEXT J,I,K:GOTO 260
```

SUPERBE ASTUCE POUR CREER UNE VALEUR ALEATOIRE

```
20 MODE 1
30 LOCATE 10,10:PRINT PEEK(&B8B4)
40 GOTO 30
```

PETIT PROG POUR AFFICHIER VOS ECRANS. L'ECRAN DOIT ETRE SUR LA DISQUETTE ET PORTER LE NOM "ECRAN.BIN".

```
20 MODE 2:INK 1,0:INK 0,0:BORDER 0:LOAD"
ecran.bin",&C000
30 b=1:a$="2100C0ED5F77237CB720F80E0A210
0C01100401ABE2813477E3006C632300A1804C63
23803B83801787713237CB720E30DC20D80C9000
0002100C01100400100040EDB0C9":FOR a=&8000
 TO &8000+67:POKE a,VAL("&"+MID$(a$,a-&8
000+1*b,2)):b=b+1:NEXT:CALL &8036
40 MODE 0
50 ' Mettre les bonnes couleurs ici
60 CALL &8000
```

GENIAL ! DESSINER EN UN RIEN DE TEMPS DE SUPERBES PAYSAGE DESSERTIQUE. MAGNIFIQUE !

```
20 DIM A(250),B(250),C(250)
30 MODE 0:NM=6:AM=30:INK 0,11
40 FOR I=0 TO 15:INK I,RND*26
50 NEXT:FOR N=1 TO NM
60 A(N)=RND*80/N:B(N)=RND*2*PI
70 C(N)=RND*15:NEXT
80 PRINT:PRINT "     DESERT PRIVE"
90 FOR X=0 TO 639 STEP 4
100 PLOT X,0,0:Y=10
110 K=2*PI*X/640:FOR N=1 TO NM
120 Y=Y+A(N)*(1+SIN(N*K+B(N)))
130 DRAW X,Y,C(N):NEXT N,X
140 FOR I=1 TO 3000:NEXT:RUN
```

VIBREZ AVEC NOUS

```
20 SPEED INK 100,100:MODE 0:BORDER 0:INK
 0,0:INK 1,26,2:OUT &BC00,8:OUT &BD00,1:
PRINT:PRINT:PRINT:PRINT:PRINT:PRINT"    S
i je tremble :                          C
'est de froid.":CALL &BB06
```

POUR TOUT SAVOIR SUR L'ETAT DE VOS DISQUETTES DANS LE LECTEUR

```
20 MODE 1:BORDER 0:PRINT:PRINT"     Etat
du lecteur de disquette"
30 OUT (&FA7E),1
40 FOR I=1 TO 999:NEXT
50 OUT (&FB7F),4
60 OUT (&FB7F),2-PEEK(&A700)
70 DK=INP(&FB7F)
80 OUT (&FA7E),0
90 IF (DK AND 32)=0 THEN PRINT:PRINT"
Le lecteur de disquette est vide":GOTO 1
20
100 IF (DK AND 64) THEN PRINT:PRINT "
   La disquette est protegee.":GOTO 120
110 PRINT:PRINT"      tout est OK !!!"
120 END
```

PERMUTTEZ DEUX ECRAN SANS LES CHARGER

```
20 MODE 0:FOR i=&3000 TO &3004:READ a:PO
KE i,a:NEXT
30 DATA &3E,&40,&C3,&08,&BC
40 FOR L=1 TO 100:DRAW INT(RND(1)*640+1)
,INT(RND(1)*400+1),INT(RND(1)*16):NEXT:C
ALL &3000:CLS:FOR L=1 TO 100:DRAW INT(RN
D(1)*640+1),INT(RND(1)*400+1),INT(RND(1)
*16):NEXT
50 A$=INKEY$:IF A$="" THEN 50
60 OUT &BC00,12
70 OUT &BD00,&30
80 OUT &BC00,13
90 OUT &BD00,0
100 A$=INKEY$:IF A$="" THEN 100
110 OUT &BC00,12

120 OUT &BD00,&10
130 OUT &BC00,13
140 OUT &BD00,0
150 GOTO 50
```

FAIRE SEMBLANT D'ETRE UNE BETE COTE RASTERS (COMME UN PRO)

```
10 OUT &BC00,1:OUT &BD00,60:OUT &BC00,2:
OUT &BD00,50
20 ON BREAK GOSUB 170
30 DATA 1,2,14,20,26,20,14,2
40 DATA 0,192,12,204,48,240,60,252
50 FOR I=1 TO 8:READ A:INK I,A
60 A(I)=A:NEXT I:BORDER 0:INK 0,0
70 ADR=50832:MODE 0:SPEED INK 1,1:INK 9,
13,26:INK 10,1:PEN 9:PAPER 10
80 FOR K=0 TO 1:RESTORE 40:FOR I=0 TO 7
90 LOCATE 1,19:READ IN:FOR J=0 TO 93
100 POKE ADR+I*2048+J+120*K,IN
110 POKE ADR+I*2048+J+120*K-600,IN
120 NEXT J,I,K
130 PAPER 0:PEN 1:PRINT " Pour un LOOK d
e pro !! "
140 A=A(1):FOR I=1 TO 7:A(I)=A(I+1):INK
I,A(I)
150 NEXT I:A(8)=A:INK 8,A(8)
160 GOTO 140
170  OUT &BC00,1:OUT &BD00,40:OUT &BC00,
2:OUT &BD00,46:MODE 1:PAPER 0
```

```
100 ENV 1,1,12,1,1,-1,6,10,-1,8:ENV 2,1
,10,1,2,-1,4
110 ENT 3,2,6,1,1,-12,1
120 ENV 3,5,0,1,10,-1,11
130 ENT 1,55,-2,1:ENT -2,2,1,6,4,-1,2,2
,1,6
140 DATA 2025,1012,506,506,426,338,2025
,1012,506,506,426,319,2025
150 DATA 1012,506,506,426,338,506,426,3
38,2273,1136,568,568,451,379
160 DATA 1012,676,1012,638,1012,676,676
,1136,758
170 RESTORE 140:FOR I=1 TO 9:READ N1(I)
,N2(I),N3(I):T(I)=40:NEXT
180 RESTORE 160:FOR I=1 TO 9:READ N(I):
NEXT
190 T(6)=25:T(7)=15
200 a1=1:a2=9:GOSUB 470:GOSUB 470
210 DATA 63,71,84,71,63,71,84,89,95,106
,127,106,95,127,89,84
220 DATA 71,84,95,89,84,89,95,106,95,10
6,127,142,169,142,127,106
230 DATA 127,106,95,127,89,127,84,71,84
,95,89,84,71,63,71,84
240 DATA 71,63,53,63,71,63,71,84,89,84,
89,95,106,96,106,127
250 DATA 71,84,63,84,53,84,71,53,56,80,
63,80,53,80,63,53
260 DATA 84,127,89,127,95,169,142,127,1
13,142,95,113,71,95,56,71
270 DATA 71,84,63,84,53,84,71,53,56,80,
63,80,53,80,63,53
280 DATA 84,127,89,127,95,169,142,127,8
9,95,106,127,169,127,63,127
290 DATA 95,106,127,142,127,106,95,47,2
4,47,95,190,95,47,45,47
300 DATA 106,127,95,127,106,127,95,127,
89,127,95,127,106,127,95,63
310 DATA 95,106,127,106,95,89,95,106,95
,106,127,142,169,142,127,106
320 DATA 127,142,169,179,190,179,169,14
2,169,179,190,213,190,213,253,284
330 ET=0:FOR i=1 TO 2
340 RESTORE 210:GOSUB 420:GOSUB 470
350 RESTORE 230:GOSUB 420:GOSUB 470
360 NEXT i:RESTORE 250:FOR p=1 TO 2:ET=
3:GOSUB 420:GOSUB 420:NEXT p
370 a1=1:a2=2:GOSUB 470:SOUND 1,213,160
,12,3,2
380 SOUND 6,216,55,10,,1:SOUND 6,106,10
5,10,3,2
390 a1=3:a2=4:GOSUB 470:SOUND 1,106,160
,12,3,2
```

```
100 ENV 1,1,12,1,1,-1,6,10,-1,8:ENV 2,1
,10,1,2,-1,4
110 ENT 3,2,6,1,1,-12,1
120 ENV 3,5,0,1,10,-1,11
130 ENT 1,55,-2,1:ENT -2,2,1,6,4,-1,2,2
,1,6
140 DATA 2025,1012,506,506,426,338,2025
,1012,506,506,426,319,2025
150 DATA 1012,506,506,426,338,506,426,3
38,2273,1136,568,568,451,379
160 DATA 1012,676,1012,638,1012,676,676
,1136,758
170 RESTORE 140:FOR I=1 TO 9:READ N1(I)
,N2(I),N3(I):T(I)=40:NEXT
180 RESTORE 160:FOR I=1 TO 9:READ N(I):
NEXT
190 T(6)=25:T(7)=15
200 a1=1:a2=9:GOSUB 470:GOSUB 470
210 DATA 63,71,84,71,63,71,84,89,95,106
,127,106,95,127,89,84
220 DATA 71,84,95,89,84,89,95,106,95,10
6,127,142,169,142,127,106
230 DATA 127,106,95,127,89,127,84,71,84
,95,89,84,71,63,71,84
240 DATA 71,63,53,63,71,63,71,84,89,84,
89,95,106,96,106,127
250 DATA 71,84,63,84,53,84,71,53,56,80,
63,80,53,80,63,53
260 DATA 84,127,89,127,95,169,142,127,1
13,142,95,113,71,95,56,71
270 DATA 71,84,63,84,53,84,71,53,56,80,
63,80,53,80,63,53
280 DATA 84,127,89,127,95,169,142,127,8
9,95,106,127,169,127,63,127
290 DATA 95,106,127,142,127,106,95,47,2
4,47,95,190,95,47,45,47
300 DATA 106,127,95,127,106,127,95,127,
89,127,95,127,106,127,95,63
310 DATA 95,106,127,106,95,89,95,106,95
,106,127,142,169,142,127,106
320 DATA 127,142,169,179,190,179,169,14
2,169,179,190,213,190,213,253,284
330 ET=0:FOR i=1 TO 2
340 RESTORE 210:GOSUB 420:GOSUB 470
350 RESTORE 230:GOSUB 420:GOSUB 470
360 NEXT i:RESTORE 250:FOR p=1 TO 2:ET=
3:GOSUB 420:GOSUB 420:NEXT p
370 a1=1:a2=2:GOSUB 470:SOUND 1,213,160
,12,3,2
380 SOUND 6,216,55,10,,1:SOUND 6,106,10
5,10,3,2
390 a1=3:a2=4:GOSUB 470:SOUND 1,106,160
,12,3,2
```


```
400 SOUND 6,216,55,10,,1:SOUND 6,106,10
5,10,3,2
410 GOTO 200
420 FOR jj=1 TO 9
430 SOUND 1,N(jj),T(jj),1,1:SOUND 2,N(j
j)/2,T(jj),0,1
440 IF jj=6 THEN 460
450 FOR kk=1 TO 4:READ A:SOUND 4,A,10,0
,2,et:NEXT kk
460 NEXT jj:RETURN
470 FOR ii=a1 TO a2
480 SOUND 1,n1(ii),t(ii),0,1
490 SOUND 2,n2(ii),t(ii),0,1
500 SOUND 4,n3(ii),t(ii),0,1
510 NEXT ii:RETURN
520 RETURN
```

A L'HEURE OU LES SOLUTIONS SE FONT RARES, COURAGE CRIONS

HELP

Je pense que vous savez tous que pour remplir ces 2 merveilleuses pages, il suffit de nous envoyer la ou les solution(s) des jeux que vous avez terminés. Or c'est justement là le problème. En effet, je n'ai recu que des solutions de jeux déja parues dans des numéros antérieurs ou bien trop longues pour être publiées. Donc, soyez sympa de ne marquer que l'essentiel. Maintenant, comme ces solutions existent et qu'il y a des tas de lecteurs qui nous téléphonent pour nous demander de l'aide sur ces mêmes jeux, il y aura donc, dans ce numéro, un récapitulatif des solutions déjà présentées.

Grâce à ce récapitulatif, vous pourrez noter le numéro d'*Amstrad Cent Pour Cent*, dans lequel la solution du jeu sur lequel vous calez a été publiée. Ensuite, il vous suffira de téléphoner au (16) 44 72 77 55 (c'est en province), où l'on vous donnera toutes les indications nécessaires pour pouvoir commander le ou les numéro(s) manquant(s). Facile. Pour ceux qui connaîtraient déjà ces solutions, ne perdez pas votre temps à lire ces pages, profitez-en plutôt pour nous adresser vos trouvailles. Bon maintenant, on commence .

LA CRYPTE DES MAUDITS

Je vous parlais d'une astuce pour ce jeu bien macabre. Or, il se trouve que mon texte, paru dans le numéro 46 de notre revue chérie, a subi l'affront d'une pétouille (comme un alien qui glisse sur la moquette et s'étale sur Ripley, l'héroïne du film). Notons que lors de ce lamentable incident, c'est la moquette capri.

Pour réparer la pétouille, lisez « CHARGE » qui s'était transformé, on ne sait trop pourquoi, en "CHAJQRE".

ZAP'T'BALLS

Jean-jacques Nolle nous fait un grand plaisir en nous donnant presque tous les codes de ce merveilleux jeu (ne vous inquiétez pas, l'alien en est sorti indemne, car Ripley a amorti la chute). Bon maintenant, on commence.

Le monde d'initiation :
Niveau2 : GEH
Niveau 4 : MHP
Niveau 6 : LPT
Niveau 8 : RTF
Niveau 10 : TFL
Nivau 12 : FLG
Niveau 14 : LGA
Pour finir : HLFSOKZUEARJ.

Le monde de glace :
Niveau 2 : UNB
Niveau 4 : ELI
Niveau 6 : EVA
Niveau 8 : BLE
Niveau 10 : BUT
Niveau 12 : THI
Niveau 14 : SLO
Niveau 16 : OKS
Niveau 18 : LIK
Niveau 20 : EAC
Niveau 22 : LAU
Niveau 24 : SEY
et le code final de ce monde : EAFHKIGPOBIX.

Le monde de feu :
Niveau 2 : YEM
Niveau 4 : ITS
Niveau 6 : IHT
Niveau 8 : SDR
Niveau 10 : AWK
Niveau 12: CAB
Niveau 14: TID
Niveau 16 :AER
Niveau 18 : OTE
Niveau 20 : VAH
Niveau 22 : UOY
Niveau 24 : YEH
et le code final de ce monde : BOPLHETURDAF.

Pour entrer dans le monde de rêve (Ripley est complètement sonnée, et l'alien n'est plus dans la pièce), il suffit de donner les codes des 2 mondes précédents, et voici deux codes pour *le monde de rêve* :
Niveau 2 : WAK
Niveau 4 : EUP

Maintenant, débrouillez pour les 2 ou 3 codes qui manquent. Attention ces codes sont bons pour un clavier AZERTY ! Notre Jean-Jacques gagne donc un bon d'achat de 250 F offert par Jessico. Je vous rappelle que pour recevoir ce lot, il suffit de faire une photocopie de la page, puis une de votre pièce d'identité favorite et d'envoyer le tout à l'adresse de Jessico. Bien entendu n'oubliez pas de mentionner le soft qui vous a tapé dans l'œil.

SAGA

Là c'est une aide toute simple qui vous est proposée. Pour pouvoir changer toutes les caractéristiques que vous voulez, il suffit de se munir d'une disquette formatée et de choisir vos personnages (Rypley s'est relevée furieuse et elle part à la recherche de l'alien pour se venger). Ensuite, vous devez prendre l'option « sauver vos personnages » et introduire la disquette vierge. A l'aide d'un traitement de texte tout simple et qui ne prend rien comme extension (donc 3 espaces), il suffit de lire le fichier qui sera sauvé sur le disque. Vous pourrez ainsi modifier tous les paramètres, et même les

objets, ce qui permet de prendre directement le gzougzou et de court-circuiter une grande partie du jeu Après avoir fini vos modifications, vous resauvez le fichier et vous le rechargez avec SAGA. C'etait un mégatruc offert par Pierre Roublot (Rypley ne retrouve pas l'alien et se met à pleurer... Qui va la consoler ?).

RECAPITULATIF

Nous y voici donc à cette récap, elle commence à partir du *numéro 17*, car c'est à ce numéro que débute vraiment la rubrique Help (avec les premières soluces), c'était Poum qui s'en occupait à l'époque (le reste n'était que des aides, et ça prendrait trop de place de toutes les citer).

On attaque donc par le *numéro 18*.

Exit d'Ubi Soft : : la solution d'un jeu sympa.

Sapiens : une aide pour ce superbe, mais aussi très dur jeu d'aventure.

L'Ile (le fabuleux soft d'aventure mis au point par Poum et par son ami Laurent Théron, un vrai bijou, riche et passionnant) : une petite aide de Poum pour un lecteur.

Pas de rubrique Help dans le *numéro 19*.

Donc passons au *numéro 20*.

Manoir de Mortevielle de Lankhor : l'un des meilleurs (si ce n'est le meilleur) jeux d'aventure jamais créés sur CPC.

Numéro 21

Peur sur Amityville de Ubi Soft : depuis un bon bout de temps, il était présenté dans les Petits Prix d'ACPC, là c'est la soluce qui est présentée.

Conspiration lui aussi de chez Ubi Soft : un jeu avec de bons graphismes et toujours la soluce.

Numéro 22, plein de trucs pour les jeux suivants.

Arkanoïd : l'un des meilleurs casse-briques sur CPC (le meilleur étant Titan et personne ne me contredira).

Bruce Lee : ce jeu assez mal réalisé, moche même, pourrait faire un très bon listing, m'enfin y'en a qui aiment (berk !)

Grand Simulateur 2 : un des nombreux « budgets » de l'éditeur anglais Code Master.

Captain America : la solution complète de ce jeu.

Crash Garret : comme pour Captain America, la solution.

Numéro 23

Stormlord : la solution des 2 premiers niveaux, de cet excellent jeu.

Le Passager du Temps de Ere Informatique : solution de la première partie uniquement.

Numéro 24

Passager du Temps : la suite de la soluce du *numéro 23*.

A320 : la solution complète de jeu mi-aventure/mi-simulateur.

Numéro 25

Heroes of the Lance d'US Gold : un plan pour ce Donjon & Dragons version CPC.

La Chose de Grotembourg : encore et toujours la solution.

Profession détective : un jeu que personne à la rédac ne connaît.

After the War : le code du second niveau, mais je vous le donne maintenant (vous n'allez pas commander un numéro spécialement pour ça : 94656981).

Numéro 26

Rick Dangerous : un numéro spécial pour ce jeu avec le plan du troisième niveau, des bidouilles dans la rubrique Pokes, il y avait aussi la solution du level 1 et 2.

Numéro 27

Jaws de MBC : la solution d'un jeu qui était trop dur, mais très bien réalisé tout de même.

Alphakhor de Loriciel : encore un bon jeu (c'est normal, on est sur CPC).

Sky Hunter : encore un bon jeu, si on tient compte de la date de sa sortie.

Pas de rubrique dans le *numéro 28* donc :

Numéro 29

Defender of the Crown d'Ubi Soft : la solution de ce jeu par les auteurs d'Exit.

Mike et Moko : encore de MBC et encore trop dur.

Han d'Islande : un jeu assez vieux, il y a tout de même la solution.

Eagles Rider : des trucs et astuces.

Manoir de Mortevielle : les codes, pour le blabla voir plus haut.

Numéro 30

Satan : le plan du 2e level de ce jeu très coloré.

Oxphar : des trucs pour ce jeu testé dans un des premiers numéros d'*ACPC*.

Iron Lord : il a obtenu 98 %, ce qui montre la qualité du jeu, alors si vous êtes perdu dedans, il y a plein d'astuces !

Numéro 31

Targhan : la solution de ce jeu fait par l'auteur de Titan, c'est-à-dire, Philippe Pamard.

La Secte noire : la solution de ce jeu, les auteurs ont depuis fait Mokowé et la Crypte des Maudits.

Navy Moves : la solution de la seconde partie de ce merveilleux jeu (mais vraiment trop dur).

Iron Lord : un truc pour ce super jeu (j'en ai parlé un peu plus haut).

Ghouls'n'Ghost : un truc pour avoir une armure magique au premier niveau.

Numéro 32

La Secte noire : la suite de la solution.

Sram 2 d'Ere Informatique : la solution de ce jeu trop facile, était-ce vraiment la peine ?

Barbarian : des trucs pour décapiter l'ordinateur sans problème.

Bloodwych : un cheat mode pour ce jeu merveilleux.

Pipe Mania : les huit codes pour les niveaux, les voici, car ce n'est pas très long, fine, news, fail, sail, eric, tape, slow, ache.

Numéro 33

Rick Dangerous 2 : la solution du premier niveau.

Renegade 3 : un petit truc.

Exolon : encore un truc, tapez ZORBA dans le redefine keys (pour un clavier QWERTY).

Bumpy de Loriciel : (il faut presser sur A, Z, Z, Q, S pour se téléporter dans les levels compris entre 1 et 100).

Harry & Harry : la solution d'un jeu que je ne connais pas.

Numéro 34

Maraudeur d'Ubi Soft : la solution. Savez-vous que ce jeu a été réalisé à partir des routines d'Aventure avant tout, vous savez la rubrique où Poum (encore lui) expliquait comment créer un jeu d'aventure. Si vous voulez absolument réutiliser ces merveilleuses routines toutes simples, il vous suffira de commander les numéros d'*Amstrad Cent Pour Cent* 13 à 19

Sram : la solution étant trop évidente, il n'y avait que quelques trucs.

A320 : quelques trucs pour ce jeu (j'en parle plus haut).

Castel Master : encore quelques trucs pour un fabuleux jeu.

Masque d'Ubi Soft : je ne connais pas ce jeu, mais je peux vous dire qu'il y a quelques astuces.

Voila c'est fini, et n'oubliez pas la suite au prochain numéro, car sinon l'alien va venir vous raser le crâne à la Ripley et M. Kruger (Freddy pour les intimes) hantera vos nuits avec toutes ses âneries (cela m'étonnerait qu'une nuit suffise). Bon j'arrête, car je deviens gaga à force de me relire .

Ludotronic, qui ne sait avec quel entonnoir se couvrir



LES DERNIERS MYSTERES

Oui, je sais ! Il n'y a pas eu d'article sur le CPC Plus dans le numéro précédent, mais c'est la faute de Pict qui a pris toute la place.

Hmmm, après cette explication fumeuse, venons-en directement au sujet de la prose de ce mois. Cet article met enfin un terme à la description de tous les registres « secrets » du CPC Plus.

Je vais donc aborder :

- les interruptions du CPC+ ;
- la rupture facile ;
- la gestion des ports analogiques ;
- la gestion du 8ᵉ bit de l'imprimante.

LES INTERRUPTIONS !

Argh ! Nous avons déjà abordé le chapitre des interruptions dans la rubrique Logon (*cf. Amstrad CPC* n°33, p. 50) et Digit vous avait tout expliqué. Pour reprendre, une interruption est le nom donné à une action (le plus souvent un branchement) effectuée par le microprocesseur (en l'occurrence le Z80A). Celui-ci interrompt l'exécution séquentielle des instructions pour simuler un appel à une routine (un CALL en Z80A si vous préférez). Deux questions se posent dès lors.

Qui dit au micro-processeur qu'il va devoir « interrompre » le programme en cours ?

Tout simplement un périphérique (par exemple un circuit ou le microprocesseur lui-même [dans certains cas, mais c'est une autre histoire]). Sur le CPC en mode « normal », les interruptions sont émises par le circuit vidéo. Celles-ci sont périodiques et sont donc fonction du temps. Le Z80A est interrompu tous les 1/300ᵉ de seconde (soit 6 fois durant 1 balayage, sachant que la première a lieu en même temps que le début du VBL).

A quelle adresse le Z80A va-t-il ?

A cette question... il y a plusieurs réponses. Ben pourquoi donc ? Parce qu'il y a les interruptions ordinaires sur Z80A (modes IM0, IM1 et IM2), les demandes de Bus (BUSRQ) et les interruptions non masquables (NMI).

Je signale au passage que je parlerai ici uniquement des interruptions ordinaires (INT), mais sachez néanmoins que la NMI est générée par le bouton poussoir de votre multiface 2 (le vecteur de cette interruption se trouvant en #0066). Ah OK, mais pourquoi ?

Le mode IM0 correspond aux instructions Z80A liées aux appels d'interruption et ne nous intéresse pas.

Le mode IM1 correspond à un mode non vectorisé et le mode IM2 à un mode vectorisé des interruptions.

La vectorisation attribue à un périphérique (souvenez-vous, c'est un déclencheur d'interruption) une adresse bien précise.

En mode non vectorisé, il y a une seule adresse (située en #0038) et la routine située à cet endroit doit alors tester quel périphérique l'a appelée.

Sur le CPC+ en mode « old generation », seul le Crtc émulé « génère » des interruptions. Il n'y a donc pas de problème pour deviner de laquelle il s'agit. De ce fait, le mode vectorisé IM2 ne sert à rien, si ce n'est de permettre le changement de l'adresse « immuable » #0038 pour les interruptions, mais les bugs de l'IM2 sur les vieux CPC obligent une « dépense » de 256 octets de Ram pour avoir une autre adresse d'interruption.

VECTORISER EN MODE IM2

Pour vectoriser en mode IM2, il faut utiliser une table de vecteurs (entendez « adresse d'interruption » pour « vecteur »). Ainsi, dans une table se suivent des adresses de routines d'interruption.

Comment définir l'adresse de la table d'interruption ?

a) L'octet de poids fort de l'adresse de la table est défini par le registre I du Z80A (mais si, mais si, il servait à quelque chose !).

b) L'octet de poids faible de l'adresse est normalement fourni par le périphérique qui génère l'interruption. Le problème sur les vieux CPC est que, ce mode ne servant pas, les 7 bits de poids fort de cette valeur sont aléatoires (en outre, le bit 0 est étrangement à 1 sur les « old » et à 0 sur le CPC+, d'où une incompatibilité du mode IM2 entre les 2 machines !) C'est pourquoi il est nécessaire de recopier 128 fois la même adresse d'interruption pour utiliser le mode IM2 sur les vieux CPC, ceci à partir de : « adresse *modulo* 256 » + 1 (pour le bit 0).

Mais revenons au Plus. L'Asic, qui est LE périphérique principal du Z80A, est capable de générer 4 interruptions différentes :

- une interruption raster ;
- trois interruptions DMA-son.

Le schéma n°1 décrit l'adresse de la table des vecteurs sur CPC+. L'octet situé en #6805 Asic (IVR) permet de définir, avec le registre I, les bits 3 à 15 de l'adresse de la table. Lorsqu'une interruption a lieu, les bits 1 & 2 sont positionnés par le périphérique qui génère l'interruption, d'où branchement à une adresse spécifique. Notons au passage que le bit 0 de l'adresse vaut toujours 0 et implique que la table commence sur une adresse paire.

En outre, la grande précision au niveau de l'adresse apportée par le registre IVR permet de ne « dépenser » que 8 octets pour la table d'interruption.

En mode IM1, nous avons vu que les interruptions ont toutes lieu en #0038. La routine située à cette adresse doit donc être capable de savoir quelle était l'interruption générée par l'Asic. Pour cela, il existe un registre de statuts dans l'Asic, qui permet de savoir quelle interruption a eu lieu. Vous connaissez déjà ce registre, appelé DCSR, car je l'avais décrit dans le n°45, page 39, schéma 3. Le revoici sur le schéma n°2.

Que l'interruption ait bien lieu est une chose, mais il faut savoir dire qu'elle est finie pour qu'une nouvelle puisse « arriver ». On appelle cette action un acquittement d'interruption.

D'ordinaire, un simple EI suffit à informer le Z80A qu'il pourra accepter une nouvelle interruption. Sur le CPC+, plusieurs méthodes s'offrent à nous.

CPC +

Schéma 1
Calcul de l'adresse d'un vecteur

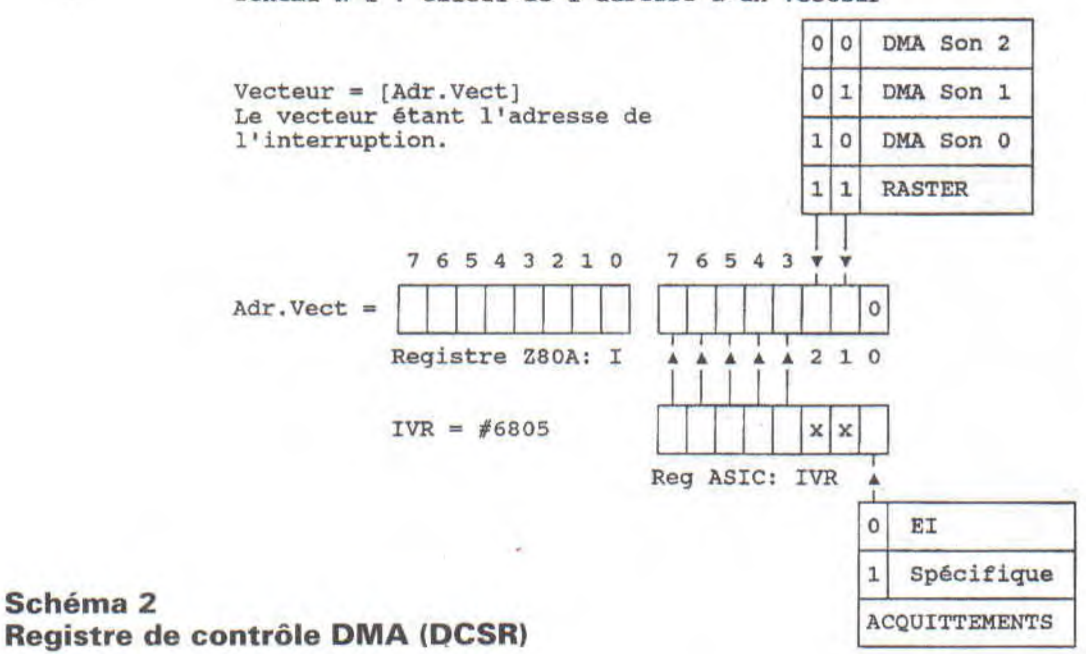


Schéma 2
Registre de contrôle DMA (DCSR)

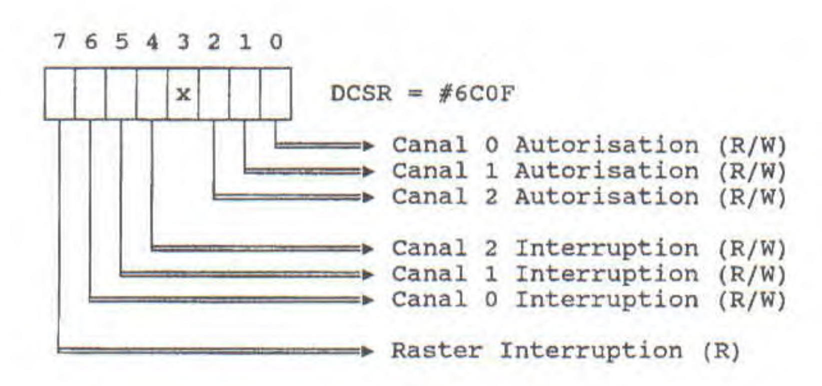


Schéma 3

Inter.	Acquittement interruption	
RASTER	(a) EI (b) Bit 4 à 1 de MRER OUT #7F00, 100xxxxx	
	Bit 0 IVR = 1	Bit 0 IVR = 0
DMA SON 2 DMA SON 1 DMA SON 0	Bit 4 DCSR = 1 Bit 5 DCSR = 1 Bit 6 DCSR = 1	EI EI EI

Schéma 4
Convertisseur analogique/digital

```
Schéma N°4 : Convertisseur Analogique/Digital

 7 6 5 4 3 2 1 0
|x|B|v|v|v|v|v|v|   ADC0 = #6808
|x|B|v|v|v|v|v|v|   ADC1 = #6809
|x|B|v|v|v|v|v|v|   ADC2 = #680A
|x|B|v|v|v|v|v|v|   ADC3 = #680B
|x|x|x|x|x|x|x|x|   ADC4 = #680C (Non connecté)
|x|x|x|x|x|x|x|x|   ADC5 = #680D (Non connecté)
|x|x|x|x|x|x|x|x|   ADC6 = #680E (Non connecté)
|x|x|x|x|x|x|x|x|   ADC7 = #680F (Non connecté)

B : Etat 1/0 selon broches 2, 6, 10 & 14 du connecteur.
v : Valeur résistance à l'entrée (De #00 à #3F)
x : Non cablé

Brochage Prise Analogique (15 broches femelle type D) :

                          Correspondance avec Joystick
                          Analogique Std sur PC.
01 : Masse
02 : B (ADC0)             Fire 1 Joystick 1
03 : v (ADC0)             Position X Joystick 1
04 : COM1 (Switche)
05 : +5 V
06 : v (ADC1)             Position Y Joystick 1
07 : B (ADC1)             Fire 2 Joystick 1
08 : Masse
09 : Masse
10 : B (ADC2)             Fire 1 Joystick 2
11 : v (ADC2)             Position X Joystick 2
12 : COM2 (Switche)
13 : v (ADC3)             Position Y Joystick 2
14 : B (ADC3)             Fire 2 Joystick 2
15 : Masse
```

Le tableau du schéma n°3 décrit tous les acquittements possibles.

Signalons au passage que le bit 0 du registre IVR, placé à 0, permet d'utiliser EI comme instruction d'acquittement automatique non spécifique à une interruption (ceci pour les DMA puisque l'interruption raster est toujours acquittée automatiquement).

Ceci a été fait pour faciliter les acquittements en mode vectorisé (IM2), puisque chaque interruption a sa routine propre.

L'inconvénient de ce procédé (bit 0 de IVR à 0) en mode IM1 est que les bits 4 à 7 de DCSR ne sont plus mis à jour (ou du moins testables), et la routine ne peut donc pas savoir quelle interruption a eu lieu.

POUR RESUMER

En mode IM2, vous savez implicitement quelle interruption a lieu et vous pouvez vous contenter d'acquitter vos interruptions par EI en plaçant le bit 0 de IVR à 0.

En mode IM1, vous pouvez :

- connaître quelle interruption a eu lieu grâce à DCSR, auquel cas le bit 0 de IVR est à 1, et vous devez acquitter votre interruption en positionnant le bit correspondant dans DCSR à 1. Notons toutefois que la « consultation » de DCSR doit impérativement commencer par l'interruption raster, car celle-ci est acquittée automatiquement et de ce fait classée prioritaire !
- ne pas connaître quelle interruption a eu lieu, mais l'acquitter automatiquement grâce à EI en positionnant le bit 0 de IVR à 0.

Sachez néanmoins que la valeur du bit 0 de IVR au reset est 1.

LES DIFFERENTES INTERRUPTIONS

L'interruption Raster permet de déclencher une interruption à une ligne spécifiée à l'adresse #6800 (PRI).
Il est possible de reprogrammer cette interruption pour générer plusieurs interruptions durant le balayage. Notons que la ligne prise en compte est la première ligne de données affichées et non pas la première ligne du balayage !
Ceci peut être gênant pour avoir des interruptions « en dehors de l'espace vidéo ». Lorsque le contenu de #6800 est différent de 0, alors l'interruption raster remplace l'ancien système d'interruptions émulé.
Ainsi vous avez peut-être remarqué qu'une valeur différente de 0 en #6800 de la page Asic provoque un ralentissement général du CPC, et pour cause : il n'y a plus qu'une interruption par balayage au lieu de six !

LES INTERRUPTIONS DMA-SONS

Hem ! Celles-ci sont générées *via* les AY-Listes que je vous ai décrites dans le n°45, pages 38-40.
L'instruction INT (#4010) lue par un canal DMA dans une AY-Liste permet de déclencher une interruption. Vu la taille de l'article sur le son, je vous conseille de le relire !
Voilà, fini le tour d'horizon des interruptions du CPC+ ! Pfewww !

LA RUPTURE

Passons maintenant à la rupture sur le CPC+. Cette technique, superbidouillage des anciens CPC, est maintenant prévue d'origine dans les entrailles de l'Asic (qui émule le CRTC !).
Il suffit juste de spécifier le numéro de ligne et l'adresse du nouvel écran ! Ainsi (#6801 ou Reg SPLT) contient le numéro de la ligne de rupture. Et (#6802, #6803 ou Reg SSA) contient l'adresse du second écran (le premier étant défini par les classiques registres 12 et 13 du CRTC émulé).
Tout comme pour l'interruption raster, le numéro de ligne varie entre 1 et 255 (0 pour annuler le « split ») et commence à partir de la première ligne de données affichées. Ceci permet ainsi de prendre en compte l'écran sur toute sa hauteur. L'overscan vertical étant de 240 pixels environ (contre 312 pour l'ensemble du balayage).
Après ce bref interlude dans le merveilleux monde de la rupture, passons à la prise située sous l'interrupteur (mais si, elle sert à quelque chose !).
Cette prise est une entrée analogique composée d'un « convertisseur analogique/digital octal en conjonction avec un réseau R-2R, un comparateur et un multiplexeur analogique ». A vos souhaits !
A quoi ça sert ?
A beaucoup de choses, mais, entre autres, à brancher 4 paddles, 2 joysticks analogiques, etc. On aurait pu penser au son si la vitesse d'échantillonnage avait été autre.
Un paddle est, en quelque sorte, un potentiomètre permettant de déplacer unidirectionnellement la raquette de votre casse-briques, par exemple. Quant à définir un joystick analogique par rapport à un joystick digital, disons que c'est un joystick « mou » (Poum est un spécialiste de la chose [NDPoum : Longshot l'a testé pour vous]) disposant d'une très haute précision et équipé de deux poussoirs (et pas de malentendus).
L'Asic dispose de 8 canaux d'entrée analogique, mais seulement 4 parmi eux sont reliés à la prise physiquement (*no comment* sur le gaspillage).
Bref, 8 registres existent, dont 4 sont lisibles. Ces registres sont des registres 6 bits mis à jour 200 fois par seconde et traduisent des variations de 0 volt (valeur #00) à 2,5 volts (valeur #3F). Notez que l'impédance en entrée doit être de 1 kilo-ohm.
Cependant, chers lecteurs, la doc technique Amstrad, largement incomplète, ne précise pas où les boutons « Fire » sont gérés. Je suppose donc, n'ayant pu le tester à ce jour, que le bit 6 de chaque registre contient l'état du Fire (ou du moins un état fourni sur les broches correspondantes aux Fires).
Je vous livre dans le schéma n°4 la structure de ces registres, ainsi que le brochage de la prise pour les joysticks analogiques compatibles PC200 (PC-8).

ET L'IMPRIMANTE ?

Et pour finir en beauté, si nous parlions de l'imprimante. Pour lui envoyer un octet, il suffit de lui demander si elle est prête, ceci *via* le bit 6 du port B du PPI situé en #F500 (*cf.* n°43, pages 23 à 27).
Lorsque ce bit est à 0, on peut lui envoyer un octet. Lorsqu'il est à 1, l'imprimante est occupée (son buffer est plein, il manque du papier, elle est éteinte, pas OnLine, ...).
Notons que le système arrive à faire la différence entre un buffer plein et un autre type d'erreur lorsque le bit de contrôle reste trop longtemps à l'état 1.
Jusqu'à maintenant, le fonctionnement de l'imprimante me paraissait simple. Il suffisait pour lui envoyer un octet d'utiliser le port #EF00 en envoyant 3 fois la valeur (1re : Bit 7 = 0 ; 2e : Bit 7 = 1 ; 3e : Bit 7 = 0), le bit 7 servant à valider l'octet ! Pour cette raison, seuls les 7 bits de poids faibles étaient pris en compte. C'est toujours le cas sur ce port pour des raisons de compatibilité !

Mais alors ? Où est le 8e bit manquant ? (NDLR : Oui, où ?) Accrochez-vous. Celui-ci se trouve sur le bit 3 du registre 12 du CRTC 6845 émulé (*cf.* n°38, page 48).

Comme quoi, il n'y a pas de petites économies, puisque ce bit « semblait » inoccupé pour les techniciens d'Amstrad. Ceux-ci ne connaissaient malheureusement pas les Overscan-Bits.

Dans le cas d'un logiciel overscan faisant des impressions, le buffer vidéo fera la valse entre 16 K et 32 K sans arrêt (très psychédélique). Heureusement toutefois que la rupture est là !

Bref, pour envoyer un octet 8 bits à l'imprimante, voici quelques lignes de basic que vous adapterez sans problème en assembleur si vous souhaitez modifier les vecteurs systèmes du firmware.

Alors, gavés ? Encore des registres ? Gaaaaa...Buurpp ! Euh... excusez-moi !

Pour les plus observateurs, vous aurez remarqué une autre prise à côté de la prise analogique (sur laquelle la doc technique originale est très incomplète. En effet, l'article est le fruit de ma sueur). Là, c'est encore mieux, la doc ne dit rien ! Et pourtant, cette prise n'est ni plus ni moins qu'une prise pour brancher un stylo optique ou même un GunPhaser (Yeah !) puisqu'il y a deux boutons de feu ! En attendant de vérifier si les registres Crtc 14 et 15 y sont pour quelque chose (j'ai un doute !), je vous donne rendez-vous pour la prochaine fois avec la carte complète de tous les registres Asic, ainsi que le brochage de TOUS les connecteurs !

Et puisque j'y suis, et pour ne pas prendre de retard voici la rectification d'une coquille dans mon article paru dans le n°45 page 38 où il fallait lire à la 8e ligne du paragraphe « Quelques remarques générales » : L'instruction STOP laisse le pointeur de la AY-Liste sur l'instruction suivante (ce qui est très pratique pour « repartir »).

Bonne nuit... Et à demain !

Longshot. Logon System 93

```
10000 REM B contient le caractère (0 à 255)
10005 REM LongshotLogon System 1993.
10010 OUT &BC00,12:A=INP(&BF00):B1=B AND 127
10020 IF ((B AND 128) = 0) THEN B2=0 ELSE B2=8
10030 A=A AND 247:OUT &BD00,A OR B2:
10040 IF ((INP(&F500) AND 64)<>0) THEN 10040
10050 OUT &EF00,B1:OUT &EF00,B1 OR 128:OUT &EF00,B1
10060 RETURN
```

AVEC THOR

Qui ne connaît pas ce dieu, si bien remis au goût du jour par les Marvels Comics ? Ce brave médecin, qui d'un coup de canne sur le sol se transforme en dieu armé d'un marteau aux pouvoirs foudroyants. Mais j'y songe, nous aussi nous avons des vecteurs qui nous rendent marteau...

La dernière fois, nous en étions restés aux vecteurs permettant la gestion des RSX. Malheureusement, il nous manquait encore quelques-uns de ces fantastiques outils pour mener à bien notre inquisition au sein du système. Cela n'est plus, car, dans ces pages, vous trouverez quelques informations complémentaires qui vont vous éclairer les points encore obscurs. Les RSX, c'est bien pratique, mais c'est mieux si on peut aussi utiliser les interruptions qui nous permettent de faire tant de jolies choses. Vous savez bien, ces trucs qui sont appelés tout seul 50 ou 300 fois par seconde. De quoi faire de belles choses résidentes et indépendantes du programme principal.

L'AVENEMENT DES EVENEMENTS

Avant tout, nous devons savoir que les vecteurs qui vont suivre utilisent des blocs d'événement qui permettent au système de travailler avec diverses routines d'interruption fonctionnant à diverses vitesses. Prenons le cas d'un bloc d'événement, comme vous en verrez de nombreux si vous utilisez cette forme de programmation. En voici la structure réduite à son plus simple effet :

octet 0 et 1 : pointeur système (il en faut)

octet 2 : compteur (simple comme compteur)

octet 3 : classe (priorité de l'événement)

octet 4 et 5 : adresse de la routine de traitement (facile)

octet 6 : adresse de sélection de la Rom (0 pour la Ram)

A ces blocs d'événement viennent se souder des blocs de contrôle qui permettent au système de partager le temps d'interruption entre toutes les routines installées, ceci sans trop ralentir le programme principal. Il en existe de deux sortes : une pour les interruptions rapides et l'autre pour les interruptions « lentes ».

LENT, TERNE ET CLAIR

Quoi que 300 ou 50 fois par seconde, on ne peut pas dire que ce soit franchement escargotesque, mais bon... Si vous avez à utiliser les interruptions rapides, soit celles émises par le CRT tous les 1/300e de seconde, il vous sera possible de retrouver un bloc de contrôle d'interruption dont la structure est simple. Les premiers composants en sont un pointeur système sur deux octets, suivis d'un bloc d'événement de la même structure que celui que nous avons décomposé dans le paragraphe précédent. Si vous vous posez les mêmes questions que moi, vous verrez que deux pointeurs systèmes se suivent. Va savoir à quoi ça sert tant de redondance. M'enfin, ils doivent bien savoir ce qu'ils en font.

Pour ce qui est des interruptions lentes (pouf pouf !), ils ont un bloc de contrôle un peu plus fourni. En voici sa bête composition :

octet 0 et 1 : pointeur système (eh oui ! encore un)

octet 2 et 3 : compteur (lorsqu'il passe à zéro, l'interruption est exécutée)

octet 4 et 5 : recharge (valeur de recharge du compteur)

Puis suit le bloc d'événement dont nous n'avons cessé de parler depuis le début.

PRECAUTIONS D'UTILISATION

Un conseil, ne perdez jamais l'adresse d'un bloc de contrôle, car vous pourrez en avoir besoin à maintes reprises. Il est aussi très intéressant de travailler sur les valeurs utilisées par ces blocs de manière à forcer la priorité de telle ou telle action. Il est de plus tellement pratique de savoir exactement où se trouvent les trucs qui influencent nos programmes. Notez aussi que, parfois, la routine d'interruption n'est pas forcément sur le plan de Rom sélectionné. Dans ce cas, il arrive que le système décide de ne pas lancer votre petit programme en attendant de meilleures conditions pour le faire. Il vaut mieux, alors, déplacer la routine d'interruption dans les 32 Ko de mémoire centrale, ce qui permet de la voir lancée presque à tous les coups, si le niveau de priorité est bien entendu assez fort. Analysez bien les blocs de contrôle. Dans la majeure partie des cas, il est possible de réinitialiser les valeurs de contrôle, ainsi que celles de priorité, pour permettre le lancement systématique de son programme. A vous de tripoter...

VECTORISEZ

Comme pour les mois précédents, nous vous donnons 4 indications par vecteur passé en revue :

1 - l'adresse d'appel (indispensable voire primordiale) ;

2 - un bref commentaire sur la raison d'être (ce qu'il fait) ;

3 - les conditions d'appel passées dans les registres ;

4 - les conditions finales, soit les résultats de l'action émise ainsi que les registres ou zones modifiés.

En voiture Thor, et sans tarder.

• BCD7 : initialisation et dépose d'un bloc d'événement dans la liste de ceux à activer lors d'une interruption en provenance du CRT

Lorsque vous appelez ce vecteur, il met tout en ordre pour que la routine citée soit appelée à chaque fois qu'il est possible.

CA : contient l'adresse du bloc d'événement ;
B contient la classe de l'événement ;
C contient l'adresse de sélection de la Rom ;
DE contient l'adresse de la routine à insérer dans la file des événements temporisés.

CF : AF, DE et HL modifiés.

• BCDA : dépose d'un bloc d'événement dans la liste de ceux à activer lors d'une interruption en provenance du CRT

Comme il est possible d'enlever un bloc d'événement de la file d'attente avec le vecteur suivant, il est possible de le remettre avec celui-ci.

CA : HL contient l'adresse du bloc d'événement ;

CF : AF, DE et HL modifiés.

• BCDD : enlève un bloc d'événement de la liste de ceux à activer lors d'une interruption en provenance du CRT

Voir la légende du vecteur précédent.

CA : HL contient l'adresse du bloc d'événement ;

CF : AF, DE et HL modifiés.

• BCE0 : initialisation et dépose d'un bloc d'événement dans la liste de ceux à activer lors d'une interruption rapide (tous les 1/300e seconde)

S'il te plaît, monsieur le système, peux-tu me lancer cette routine 300 fois par seconde ? Pas de problème Batman, on y va...

CA : HL contient l'adresse du bloc d'événement ;
B contient la classe de l'événement ;
C contient l'adresse de sélection de la Rom ;
DE contient l'adresse de la routine à insérer dans la file des événements temporisés.

CF : AF, DE et HL modifiés.

• BCE3 : pose d'un bloc d'événement dans la liste de ceux à activer lors d'une interruption rapide

Le même vecteur que &BCDA mais pour les interruptions rapides.

CA : HL contient l'adresse du bloc d'événement.

CF : AF, DE et HL modifiés.

• BCE6 : enlève un bloc d'événement de la liste de ceux à activer lors d'une interruption rapide

Comme &BCDD mais pour les interruptions au 1/300e

CA : HL contient l'adresse du bloc d'événement.

CF : AF, DE et HL modifiés.

• BCE9 : dépose simple d'un bloc d'événement dans la liste de ceux à activer lors d'une interruption normale (tous les 1/50e seconde)

Attention ! ici le bloc d'événement n'est pas initialisé. Il faut utiliser le vecteur BCEF pour cela.

CA : HL contient l'adresse du bloc d'événement ;
DE contient la valeur à installer dans le compteur ;
BC contient la valeur de recharge de ce compteur lorsqu'il atteint la valeur 0.

CF : AF, BC, DE et HL modifiés.

• BCEC : enlève un bloc d'événement dans la liste de ceux à activer lors d'une interruption normale

Il est possible de suspendre le lancement de certaines routines lentes sans pour autant les enlever de la file d'attente.

CA : HL contient l'adresse du bloc d'événement.

CF : si le bloc d'événement appartenait bien à la liste, la retenue est vraie et DE contient la valeur du compteur. Dans tous les cas, AF, DE et HL modifiés.

• BCEF : initialise un bloc d'événement de manière à ce qu'il soit gérable par les deux vecteurs précédents

Installer un événement dans une liste, c'est bien mais encore faut-il que la zone de mémoire concernée soit réellement un bloc de contrôle. C'est ici que nous le créons.

CA : HL contient l'adresse du bloc d'événement ;
B contient la classe de l'événement ;
C contient l'adresse de sélection de la Rom ;
DE contient l'adresse de la routine ;

CF : AF et DE sont modifiés et HL contient l'adresse du bloc d'événement augmentée de 7 octets.

• BCF2 : actionne un bloc d'événement

Des compteurs sont mis en action lors des appels des interruptions. Le fait de passer par ce vecteur permet le lancement conditionnel de la routine tout en modifiant les compteurs et les priorités mis en œuvre.

CA : HL contient l'adresse du bloc d'événement.

CF : les registres sauf IX et IY sont modifiés en fonction de la routine appelée.

• BCF5 : fait le ménage dans toutes les files d'attente d'événements temporisés

CA : rien à configurer.

CF : AF et HL sont modifiés.

• BCF8 : détruit un événement en l'enlevant physiquement de la file d'attente

CE : HL contient l'adresse du bloc d'événement.

CA : AF, BC, DE et HL modifiés.

• BCFB : recherche de l'événement suivant à traiter

Cela permet de savoir si des tâches restent à remplir ou non.

CA : rien à faire.

CF : si une routine est trouvée, la retenue est vraie et HL contient l'adresse du bloc d'événement. De toute manière, AF, DE et HL sont modifiés.

• BCFE : traite un bloc d'événement

Dans ce cas, le lancement de la routine associée au bloc n'est pas forcé.

CA : HL contient l'adresse du bloc d'événement.

CF : AF, BC, DE et HL modifiés.

• BD01 : termine le traitement d'un événement

Permet de modifier les niveaux de priorité des événements visés. Le système réalise cette opération en fonction des priorités de tous les événements.

• BD04 : interdiction des événements temporisés normaux

Les files d'attente d'événements ne sont plus scannées.

CA : rien.

CF : HL est modifié.

• BD07 : réautorise les événements temporisés conventionnels

Quand on a utilisé le vecteur précédent, on peut inverser l'action avec celui-ci.

CA : aucune.

CF : Hl est modifié.

• BD0A : interdit un événement

Ce vecteur permet de stopper l'action d'un événement sans pour autant perdre du temps avec la gestion de la file d'attente.

CA : HL contient l'adresse de l'événement.

CF : AF est modifié.

• BD0D : donne le temps écoulé en 1/300e de seconde depuis l'allumage du CPC

CA : rien à prévoir.

CF : DEHL forme le nombre demandé sur 32 bits.

Voilà, il ne nous reste plus qu'à passer en revue l'interfaçage avec le matériel. Et c'en est fini des interruptions. Notez que pour posséder ces dernières, rien ne vaut le fait de les utiliser à fond et de les analyser au microscope. En attendant, je m'interromps...

Sined le Barbare tabac



COLLECTOR (DE S À T)

Super bande de petits aminches, vous êtes, comme promis, au rendez-vous de la meilleure rubrique du meilleur magazine de la plus belle planète du plus parfait univers en compagnie du *meilleur* rédacteur...

Pas sur la tête, aïe ! aïe ! non pas ca ! Ok, chef, bien chef. Je disais ça, car je suis très content de retrouver mes potes. Qui plus est... Bien chef, au travail, tout de suite, dans la minute. Bien, dans la seconde. Bon d'accord, c'est vous le chef, chef. Maintenant sinon... Aïe ! non, j'y vais chef, aïe ! je suis parti, aïe ! aïe ! j'y suis déjà, c'est bientôt fini. Pfffff ! Je vais me plaindre aux syndicats, car ceci est une facon indigne de traiter le meilleur...

SENTINEL

Voici quelques codes pour avancer dans ce fabuleux jeu.

0010 : 97567465
0020 : 78042177
0030 : 41062296
0040 : 19390739
0080 : 48730556
0090 : 05480507
0100 : 57228885
0110 : 88172772
0150 : 89060253
0290 : 44457846
0520 : 29462657
0570 : 64869348
0700 : 51369747
0740 : 72678868
0880 : 97879773
0960 : 60490453
1130 : 69054842
1160 : 44999748
1390 : 60857694
1690 : 96725444
1920 : 69172992
1960 : 85898494

Avec la Multiface, je vous donne des vies à gogo.

POKE &030A,n : ou n est le nombre de points de vie que vous désirez vous attribuer.

Pour les possesseurs de Disco voici de quoi vous régaler.

Recherchez donc sous l'éditeur de Discology la chaîne hexadécimale suivante 3A,0A,03,E6,FF,28,E6,D6,01,32 et remplacez le 3A par un C9.

Voilà qui devrait vous permettre de vous amuser comme des petits fous dans les landscapes de ce jeu, qui est, répétons-le, vraiment fabuleux.

Et les autres ? Hummmmm ? Pour ceux qui travaillent avec un 464 K7, voici un petit listing.

```
1 ' INVULNERABILITE POUR SENTINEL
2 ' VERSION K7
10 MEMORY 1228:B+0:FOR I=48640 TO 48664
20 READ A:POKE I,A:B B+A:NEXT
30 IF b<>2315 THEN "ERREUR DANS LES DATAS":END
40 LOAD"SENTINEL1":CALL 48640
50 DATA 33,56,189,54,27,33
60 DATA 72,187,54,195,35,54
70 DATA 19,35,54,190,195,00
80 DATA 63,33,243,44,54,195
90 DATA 201
```

SHADOW DANCER

Avec la Multiface :

POKE &0746,&00 pour avoir des vies infinies.

POKE &07C0,&00 pour avoir de la magie infinie.

SHINOBI

Avec Disco, recherchez la chaîne hexa 8D,0D,3E,06,32,AE et remplacez le 06 par un 00. Cela évite les phases finales de chaque niveau.

Pour la Multiface, POKE &109D,&00 pour éviter les phases finales de chaque niveau. POKE &0F52,&00 pour les vies infinies. POKE &3707,&A7 pour le temps infini.

SILVA

Vous voulez de l'oxygène infini dans le labyrinthe ?

Alors « pokez 851A,00 » avec la Multiface.

SIM CITY

Pour augmenter rapidement l'argent dont vous disposez dans Sim City, chargez le jeu et formatez une disquette avec l'utilitaire de formatage du jeu. Lorsque vos finances sont au plus bas, sélectionnez le menu SYSTEM et validez l'option LOAD CITY. Un catalogue de vos cités déjà sauvegardées apparaît à l'écran. Sélectionnez un NOFILE et validez-le. Le jeu ne chargera rien, mais lorsque vous reviendrez à votre cité, vous constaterez avec joie que l'argent qui vous est dévolu s'est multiplié de façon très avantageuse.

Autre astuce avec Disco. Lancez le jeu, puis formatez une disquette. Sauvegardez une ville non créée (dans le menu SYSTEM), puis prenez discology et allez en piste 1, secteur 41, adr &0002. Mettez FF,FF,7F pour avoir 8388607$. Pour jouer, chargez votre ville et... surprise...

SKATE CRAZY

Piste 14, secteur 13, adresse &0028. Modifiez 38,04 par 00,00. Effet : permet d'obtenir des notes maximales dans la première partie.

Piste 07, secteur 15, adresse &0017. Modifiez 01 par 00. Puis piste 07, secteur 15, adresse &004F. Modifiez 01 par 00. Effet : vies infinies dans la partie arcade du jeu.

SKWEEK

Piste 21, secteur C3, adresse &00C5. Modifiez 06 par FF. Effet : 255 vies. Ou/et piste 20, secteur C2, adresse &0045. Modifiez 06 par FF. Effet : les monstres ont disparu.

SMASH TV

Recherchez la chaîne hexa B7,C2,26 et remplacez le C2 par un C3 pour être invulnérable.

Recherchez la chaîne hexa : 3D,FE,FF,CA,9C, et remplacez le 3D par un 00. Ainsi, en passant sur vos adversaires, vous les tuerez.

Avec la Multiface :

Poke &1D5C,&00 : vies infinies.
Poke &20E2,&C3 : invulnérabilité.

SOLEIL NOIR

Code d'accès au deuxième niveau : 2414520.

SOLOMON'S KEY

Recherchez 3E,05,32,05,02,3E,05,32.
Remplacez par :
3E,05,32,80,02,3E,80,32.
Et vous voilà avec 128 vies.

SORCERY

Pour avoir de l'énergie infinie, remplacez D6,01,D8,27,FD par D6,00,D8,27.

SPACE MAZES

Recherchez 3E,03,32 et remplacez par 3E,FF,32 pour 255 vies.

SPHERICAL

POKE &6A0B,&00 : potions infinies.
POKE &69DB,&3C : énergie infinie.

SPINDIZZY

Pendant le jeu, appuyez sur P pour mettre en pause, puis faites comme pour rénitialiser. A chaque fois que vous changerez de salle, vous gagnerez du temps.

SPLIT PERSONNALITIES

Piste 10, secteur 04, adresse &01AE. Modifiez C2 par C3. Puis, piste 10, secteur 05, adresse &00E4. Modifiez C2 par C3. Effet : temps infini.

Ce n'est pas tout.

Piste 09, secteur 07, adresse &0120. Modifiez 3A par 3E. Puis, piste 10, secteur 05, adresse &00B0. Modifiez 3A par 3E. Effet : vies infinies.

STARQUAKE

Les codes des stations de téléportation sont : VOREX, DULON, ASCIO (QSCIO), ELIXA (ELIXQ), ANGLE (QNGLE), ZODIA (WODIQ), SNODY, UPAZZ (UPQWW), AMBOR (Q,BOR), KRYZL (KRYWL), RALIQ (RQLIA), TALIS (TQLIS), INDOL,

POKES

RÉSUMÉ :
EN CE TEMPS-LÀ
POKES POKAIT
POUM POUMAIT
LA RÉDACTION RÉDIGEAIT
LE MINITEL MINITELAIT ...

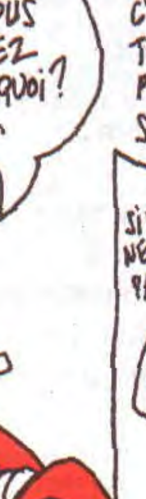
LES TEMPS SONT DURS

MAIS ON A BON ESPOIR !

... LE CHEF CHEFFAIT
MUKAÏA DESSINAIT...
EN CE TEMPS-LÀ C'ÉTAIT
LE PRINTEMPS...
TOUT LE MONDE S'EN FOUTAIT...





















POKES TROUVERA-T-IL LA BONNE SOLUTION POUR SORTIR DE CETTE CRISE ?... HEUREUSEMENT SOUS LE TITRE EN GROS IL Y A ÉCRIT "MAIS ON A BON ESPOIR !" ET L'ESPOIR FAIT VIVRE... JUSQU'AU PROCHAIN NUMÉRO... IMPATIENTS VA ! ...

OPTIK et QUORE (AUORE).

STORMLORD

Il faut frapper rapidement et sans espaces BRING ON THE GIRLS avant d'entamer une partie. Vous entendrez alors une musiquette, et les chiffres 1 et 2 apparaîtront en haut de l'écran.

Alors, vous pourrez taper sur un chiffre de 1 a 4 pour commencer le jeu au niveau correspondant avec des vies infinies.

Pour les plus honnêtes et moins exigeants. Pour avoir 255 vies, recherchez 3E,09,32,A1, et remplacez le 09 par FF.

STRIDER

POKE &2AC2,&00 : temps infini.

POKE &2BE9,&00 : vies infinies.

STRIDER 2

POKE &017E,&00 : vies infinies.

POKE &01D2,&C9 : énergie infinie.

POKE &01FB,&C9 : même effet lorsque vous êtes robot.

POKE &12E6,&A7 : temps infini.

STUNT CAR RACER

Recherchez la chaîne hexa 01,7E,32,F9,83 et remplacez par 01,7E,00,00,00 pour avoir des turbo infinis.

Recherchez la chaîne hexa FE,7E,38,07,3D, remplacez le 38 par 18 et vous serez invincible malgré les indications de l'écran.

SUPER CARS

Voici les codes de Super Cars pour les levels 2 et 3.

Class 2: ODIE

Class 3: BIGC

Sinon, recherchez la chaîne hexa 2A,E3,06,7D,93 et remplacez le 2A par un C9 pour avoir de l'argent infini.

SUPER SCRAMBLE

Recherchez 7E,D6,01,27,77 et remplacez par 7E,D6,00,27,77. Attention à la surprise !

SUPER SKWEEK

Recherchez la chaîne CB,E9,CD,5B,00 et remplacez le tout par CB,E9,00,00,00 pour avoir des vies infinies.

SWITCHBLADE

Dans le fichier CODE2.BIN, recherchez la chaîne hexa 44,0A,7E,91,28,05,FA et remplacez le 91 par un A7 pour être invulnérable.

Avec une Multiface :

POKE &30AB,&00 + POKE &30AC,&00 + POKE &30AD,&00 pour avoir des vies infinies.

TARGET RENEGADE

Piste 06, secteur 05, adresse &03CE. Modifiez 35 par B6. Effet : vies infinies.

Piste 06, secteur 05, adresse &023E. Modifiez 01 par 00. Effet : gèle le chronomètre du jeu.

TEENAGE MUTANT HERO TURTLES

En cours de jeu, tapez G,O,1 et SHIFT pour devenir invulnérable.

Avec Disco, recherchez la chaîne hexa 3D,32,7E,86,06,7F,0E,10 et remplacez le 3D par un 00 pour avoir de l'énergie infinie.

TENNIS CUP

Recherchez la chaîne 6F,3E,1E,20,02 et remplacez le 1E par FA. Vous avez désormais 255 crédits. Merci *ACPC* !

TERMINATOR 2

Voici comment commencer au niveau que vous désirez. Tout se passe en piste 7, sect 85.

Pour ne pas faire le premier niveau :

Adresse 3F, il y a CD,08,81, remplacez par 00,00,00.

Adresse 4B, il y a CD,36,81, remplacez par 00 00 00.

Adresse 51, il y a CD 66 95, remplacez par 00 00 00.

Pour ne pas faire le 2e niveau :

Comme tout à l'heure mettez trois zéros aux adresses 56,5F et 65.

Pour ne pas faire le 3e niveau :

Toujours en piste 7, secteur 85. Mettez les fameux trois zéros aux adresses 73,6A et 79

La suite logique consistera à vous éviter le quatrième niveau.

Aux adresses 7E,8A et 90, mettez ce que vous savez.

Bientôt la fin avec le 5e niveau qui peut être « skuizzé ».

Les adresses sont : 95, 9E ET A4.

Enfin pour le sixième niveau : A9,B2,B8.

Toujours pour rendre Schwarzy plus fort qu'il n'est :

- recherchez la chaîne hexadécimale 00,00,00,EA,24 et remplacez le 00 par un 01 pour avoir de l'énergie infinie.
- recherchez la chaîne hexadécimale 3E,01,CD,08,81 et remplacez le 3E,01 par 18,15 pour avoir un accès direct au 2e niveau ; par un 18,29 pour le 3e niveau ; par un 18,3D pour le 4e niveau ; par un 18,54 pour le niveau 5 ; par un 18,68 pour le niveau 6 ; et par 18,7C pour le niveau 7.

Une autre astuce consiste à entrer les lettres GEP dans le tableau des scores après avoir perdu. Vous aurez de l'énergie infinie.

Pour les fans de la Multiface :

POKE &24D5,&01 : énergie infinie.

POKE &803D,&18 : POKE &803E,&15 : départ niveau 2.

POKE &803D,&18 : POKE &803E,&29 : départ niveau 3.

POKE &803D,&18 : POKE &803E,&3D : départ niveau 4.

POKE &803D,&18 : POKE &803E,&54 : départ niveau 5.

POKE &803D,&18 : POKE &803E,&68 : départ niveau 6.

POKE &803D,&18 : POKE &803E,&7C : départ niveau 7.

POKE &A660,&00 : supprime les obstacles au niveau 2.

TETRIS

POKE &1001,&53 annule la musique du jeu.

Une autre méthode consiste à rechercher le chaîne hexa CD,37,10,C9,3E et à remplacer le 37 par un 53.

Sachez, pour les plus techniciens d'entre vous, qu'il est également possible de faire la même chose en tournant le potentiomètre de volume sonore de votre CPC.

THE DARK SIDE

Recherchez 0D,20,3C,CB,FE,EB et remplacez par 0D,20,00,CB,FE,EB pour que les ECD ne se régénèrent plus.

THE EMPIRE STRIKES BACK

Piste 07, secteur 43, adresse : &0108. Modifiez 05 par une valeur jusqu'à FF. Effet : augmente le nombre de Shields.

THE LIGHT CORRIDOR

Dans la compilation NRJ2, piste 11, secteur 01, adresse &0ACE, remplacez le 03 par un hexa supérieur et devinez la suite...

Toujours dans les surprises, POKE &3ACE,&FF avec la Multiface.

THE LIVING DAYLIGHTS

Pour 255 vies. Recherchez la chaîne hexa 3E,05,32 et remplacez le 05 par un FF.

THE REAL GHOSTBUSTERS

Tapez en même temps les touches Z, E et R et vous aurez des vies infinies. De plus, si vous retapez Z, E, R, vous traverserez l'autre niveau. Génial, non ?

THE SIMPSONS

Pour des vies infinies dans le level 1. Face 1, piste 8, secteur 82, adresse 00E4, remplacez le 01 par 00.

De même pour le 2e niveau, piste 16, secteur 89, adresse &0001, modifiez le 01 par un 00.

3e niveau. Piste 29, secteur 87, adresse &00E9, modifiez le 01 par un 00.

Sur la face 2.

Piste 05, secteur 83, adresse &0153, modifiez le 01 par un 00 pour le niveau 4.

Enfin, pour le 5e level, piste 15, secteur 86, adresse &005A, modifiez le 01 par un 00.

Avec la Multiface.

Niveau 1 :

POKE &3AE4,&00 : vies infinies.

POKE &32D6,&00 : argent infini.

POKE &15B2,&A0 : temps supplémentaire.

POKE &15A8,N : N va de 1 à 20 et correspond au nombre d'objets à peindre pour terminer le niveau.

POKE &3AC0,&C9 : invulnérabilité.

Niveau 2 :

POKE &4401,&00 : vies infinies.

POKE &1C8F,&FF : temps supplémentaire.

POKE &43DC,&C9 : invulnérabilité.

Niveau 3 :

POKE &48E9,&00 : vies infinies.

POKE &1E7C,&FF : temps supplémentaire.

POKE &48C0,&C9 : invulnérabilité.

POKE &2EF2,&00 : Bart ne chute plus.

Niveau 4 :

POKE &4753,&00 : vies infinies.

POKE &1EB0,&FF : temps supplémentaire.

POKE &472A,&C9 : invulnérabilité.

POKE &2E29,&00 : Bart ne chute plus.

Niveau 5 :

POKE &425A,&00 : vies infinies.

POKE &1D18,&FF : temps supplémentaire.

POKE &4235,&c9 : invulnérabilité.

Voici de petites astuces. Au 2e niveau, pour ne pas devoir trouver tous les chapeaux, placez-vous à la porte où il y a écrit « fruit et veg » et sautez sur les mecs qui arrivent de la porte. Au 3e niveau, après les portes, servez-vous des bulles et de la tête du clown et sautez sur la langue rouge.

Ça devrait suffire... Rendez-vous au numéro 48, et bien le bonjour chez vous.

Call x& Poke,255

La Classe DES LECTEURS

Nous revoici donc pour la sélection des meilleurs écrans de ce numéro. Avant de vous donner le nom de notre gagnant, je dois absolument vous prévenir : laissez-nous vos écrans sur Minitel. En effet, vous aurez d'autant plus de chances d'être sélectionné. Alors, par pitié ou par intérêt : des graphes !

Bon reprenons : c'est Didier Houdas qui pourra commander 750 F chez Jessico de notre part, avouez que son écran est sympa. En vous souhaitant de longues et bonnes heures devant les couleurs du CPC, nous vous donnons rendez-vous au prochain numéro, et que la CPC Force soit avec vous !



Oh, le beau volatile de Didier Houdas.



Grosse cylindrée, de L. Wahnert.

Fenêtre sur électronique de Christophe Maillot.



Le méchant bébé de Christophe Klein.



Explosion démographique de Christophe Maillot.



Le lieutenant Big Bull à fond la caisse (Anne onyme)

LE RETOUR DU PAPIER

Qu'on se le tienne pour dit : Joyeuses Pâques. Voilà, tout le monde est content ! Maintenant que les formalités ont été remplies, passons aux choses sérieuses.

Moi aussi, je suis content, les fanzines papier sont revenus en force : Youpiiiiii(iii). Allons-y !



NEW ARCADE 4 (6128 ET 6128+)

Cela faisait longtemps que j'attendais ce fanzine, qui, de par son incompatibilité avec le 6128+, n'avait pu trouver place dans le dernier numéro de *Cent Pour Cent*. Désormais, il n'y a plus de problèmes. Alors, que les fans de Schwarzy et de Stalone me suivent, car New Arcade fait partie des fanzines adorant les digitalisations de ces deux personnages, ô combien délicats... Allez, décortiquons ensemble les différentes parties qui le composent. La rubrique « pokes » vous apprendra comment bloquer le reset, accéder aux users de votre disquette, sauvegarder à la vitesse de la lumière, cacher une ligne de votre programme préféré, etc. Le tout vous est présenté avec un petit graphe, trois rasters, ainsi qu'une charmante musique : que demande le peuple ? Je ne m'attarderai pas sur les autres rubriques de cette face, car j'ai mieux, bien mieux, à vous proposer. Sur la deuxième face, rubrique « bidouilles », Epsilon vous apprend comment réduire en un fichier compacté une musique conçue sur Equinoxe, cool non ? Au menu suivant, deux jeux sont testés, il s'agit de Zap'T'Ball et Robozone. Il faut noter au passage qu'Epsilon, s'il utilise des écrans ne lui appartenant pas, s'acharne à mettre le nom de l'auteur au-dessous, une bonne initiative, que beaucoup devraient suivre... D'autre part, je dois vous apprendre un nouvelle assez triste : *New Arcade 5* ne sortira jamais. Il serait question d'un nouveau fanzine, mais là, demandez à Christophe.

Distribution NA4
76, bld du Général-de-Gaulle
78700 Conflans-Sainte-Honorine
Christophe Kastriottis (rédac')
18, rue Pierre-Curie
78700 Conflans-Sainte-Honorine

BONSOIR LA PLANETE N° 2 papier

Non, non, vous n'êtes pas sur l'émission de Sky Rock (20 h-22 h) du même nom, vous êtes bien en possession d'un fanzine CPC, et pas n'importe lequel... Allez, je ne vais pas vous faire languir plus longtemps, parlons un peu de cette merveille. C'est après quelques petits mots d'encouragement pour l'année 93, que débute la rubrique « le droit aux lecteurs ». Là, qu'ouïe-je ? La rédaction se plaint de ne pas avoir de courrier ? Je compte sur vous tous, chers CPCiens, pour remédier à ce petit problème. D'autre part, la rédaction voudrait créer une banque de données (démos, fontes, musiques, programmes). Là aussi, il est tout à fait indispensable, pour reporter au plus tard possible la mort du CPC, que tout le monde s'entraide. Seule la fraternité pourra nous sauver camarade, euh, non, je voulais dire : chers amis. Mais, revenons à nos fanzines. Les prochaines pages qui vont vous être décrites sont particulièrement intéressantes. Eh oui, il s'agit d'une initiation à l'électronique. Ici, vous apprendrez comment se constituer un bloc d'alimentation de 0 à 30 volts, un amplificateur pour votre casque de Walkman, et un œil pour votre ordinateur. Merci monsieur Robot pour cette rubrique.

Ensuite, de la page 10 à la page 14, c'est le traditionnel cours d'assembleur, qui attend, fidèle à son poste. Ainsi, au fil des numéros, vous apprendrez comment faire des vu-mètres (n'est-ce pas Ludo ?), des animations de sprites, etc. De par la place qui m'est impartie, je ne puis vous parler des autres rubriques (pokes, tests, Basic, cinéma ...) qui font de *Bonsoir la Planète* un fanzine à posséder absolument.

Moitié nord de la France
Richard Trechet
45-47, route de Tours 14700 Falaise
Moitié sud de la France
Benoît Moret
5 bis, rue du Puits-Dumay
71100 Sain-Rémy





Ce cœur restera de glace.

PIXELS N° 1 Toutes machines

Cette création n'est certes pas le chef-d'œuvre du siècle, mais *Pixels* est un des rares fanzines à exister en disquette et en cassette, cela valait le coup d'être souligné. Bien sûr, puisqu'il s'agit d'un premier numéro, il ne peut prétendre arriver au niveau d'*Arkadia* ou *Ready Fanz*, mais il présente le germe du génie qui fera de lui un des plus grands fanzines d'ici peu. Commençons par la face A. Là, un test, quoi de plus banal, eh bien ! sachez qu'il s'agit du test d'OCP Art Studio qui obtient un 97 %, à mon avis tout à fait justifié (on pourrait peut-être même mettre plus, non ?). Le listing, dans la rubrique Basic, est un mini orgue pour CPC, un petit prog tout gentil, tout mignon. Il y a encore beaucoup de rubriques, malheureusement, je vais dépasser...
Comme je vous l'ai dit, ce fanzine sera d'ici peu au top. Mais il a besoin d'aide, d'écrans, de musiques, de techniques... alors, please, aidez-le à combler ses lacunes.

Version K7 :
Frédéric Mutez
81, rue Anatole-France 62100 Calais
Version D7 :
Florent Chanteret
47, rue du Général-Giraud
62100 Calais

DEAD PLAYER 4 Papier, membre de l'AFC

Cela faisait longtemps qu'on me demandait de parler de ce fanzine (Franck Einstein, puis finalement J. Schamchula). Eh bien, je le fais ! Qu'il est beau ce fanzine, qu'il est bien fait, qu'il est marrant, mais qu'est-ce qu'il est court ! On sent bien que les rédacteurs, quitte à faire quelques pages de moins, ont décidé d'opérer dans la qualité. Bravo ! c'est une bonne initiative. Débutons le test. Dès la première page, nous apprenons la création d'une nouvelle association, qui se nomme « Factice » et qui devrait servir plus ou moins de banque de données pour fanzine. Maintenant, écrivez-leur, si vous voulez en savoir plus. La page suivante est dédiée à *Moktar*, et à *Pot de Call*. Ensuite, dans la rubrique « interview », c'est Arrakis, qui, sous le micro de Cybersoft, nous raconte, non pas sa vie, mais un peu ce qu'il veut (du délire en perspective). Enfin, ce fanzine se termine par des tests de jeux provenant de machines qui ne rivaliseront jamais avec le CPC, j'ai nommé : Zelda III (S-NES), Shinning in the Darkness (Mega D), etc. Je ne vous en dévoile pas plus. *Dead Player* est un fanzine papier à ne pas rater.

Dead Player
82, chemin du Charbonnier
69200 Vénissieux

LES NOUVELLES DU FRONT

Nous y revoilà, essayons d'être clair et précis. Zalko recherche une salle pour y organiser l'AFC Expo 2 (attention ! cette salle doit pouvoir rester ouverte tout un week-end, nuit comprise). *Ready Fanz* sera en retard (peut-être pour toujours, hélas !). *CPC Crack* recherche un collaborateur (ou collaboratrice). Contactez Marc Hufschmitt, 40 av. du Commandant-Barré, 91390 Morsang/Orge. Le nouveau *Crack'N Rom* est sorti. *Le Canard Amoché* n° 7 vient d'arriver. *CCC* n° 6 et *Exit* n° 8 viennent également de paraître. *Pot de Call* n° 4 est sorti, par ailleurs, Swab organise un meeting les 28-29 août au centre socio-culturel de Masevaux (renseignements : Pot de Call [Scham] 8, passage du Commandant-Berger 68290 Masevaux). Enfin, Totov recherche toujours « The Demo », pask, je ne l'ai jamais vue ...

Totov

Bonsoir La Planete
Fanzine Pour Amstrad
CPC 464+/6128+/6128
EDITORIAL:
LE REDACTEUR...
NUMERO 3 MOIS DE JANVIER

Qu'ils sont beaux, tous ces fanzines, et qu'ils sont faciles à commander aussi. Pour les papiers, il suffit d'un petit mot accompagné d'une enveloppe auto-adressée à 3,80F à l'adresse en bleu. Pour les disquettes, la procédure ne change guère, à la chose près, qu'il faut fournir une disquette en plus. Comme d'habitude, ces fanzines ont besoin de vous, alors envoyez-leur des bidouilles, des programmes, des musiques, des graphes, etc. Il ne me reste plus qu'à vous donner rendez-vous au prochain numéro, et d'ici là : amusez-vous bien.

Pour toutes les nouvelles sur les fanzines, le Minitel reste à votre disposition : BAL Totov.

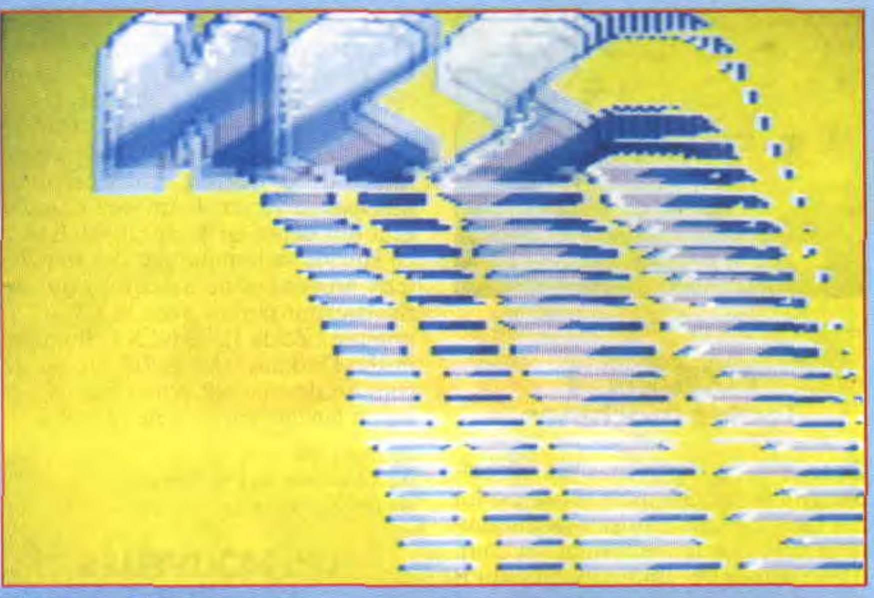
LES DEMOS HISTORIQUES

Les démos sont, depuis des années, l'arme indiscutable des programmeurs en chambre. Pour se faire reconnaître dans le milieu des CPCiens, il suffit bien souvent de faire une démo en béton.

Pour les quelques petits nouveaux dans le monde de la micro-informatique, je précise qu'une démo est un programme qui ne sert à rien, si ce n'est à vous en mettre plein la vue. Plus les techniques employées sont tordues, plus l'impact sur l'utilisateur sera grand. En gros, elles servent à glorifier un individu, ou un groupe, au sein du microcosme.
Nous avons sélectionné, dans ces pages, un échantillon représentatif de ce monde merveilleux. Notez aussi que la plupart de ces démos sont disponibles sur notre serveur Minitel 3615 ACPC.

L'ALLEMAGNE, UN PRECURSEUR

Tout commença en Allemagne avec MCS et sa « MCS Demo 3 » qui suscita un grand intérêt. Après des sphères qui montaient et descendaient en déformant le fond (un judicieux cyclage de couleurs), nous avions droit à un énorme logo MCS sautant à travers l'écran. Vous voyez que ce n'était déjà pas si mal pour l'époque, mais il fit beaucoup plus fort dans les suivantes. En effet, nous avons eu droit, pour la première fois, à des rasters. Soit, maintenant il n'y a rien d'extraordinaire à cela, mais, à l'époque, imaginez le délire de voir plus de seize couleurs dans un même logo.



MCS Demo 3.

MCS nous abreuva encore de bien d'autres bijoux, mais qui, je dois le dire, n'ont rien apporté de nouveau. Simultanément à MCS, un autre Allemand, JLCS, nous montra lui aussi quelques-unes des capacités du CPC, avec une utilisation complètement délirante du Registre 2, qui permet alors à un logo, judicieusement dessiné, de nous donner l'impression de flotter dans les airs.

LE REVEIL FRANÇAIS

C'est vrai, les premiers sur le coup étaient bien les Allemands, mais en France, si on n'a pas d'idées, on n'hésite pas à prendre le train en marche et à rattraper notre retard en bossant comme des malades.
Les premières démos honorables furent celles de Fred Crazy et du docteur TKC. Ils nous concoctèrent alors divers petits progs plus incroyables les uns que les autres. Leur dernière création, et non des moindres, fut la « Red-TKC Demo ». Cette dernière nous proposait déjà quelques techniques dont nous n'avions pas encore soupçonné la faisabilité sur nos machines. Il s'agissait bien sûr des scrolls hard. En effet, dans cette démo, le Dr TKC nous permit de découvrir un superbe scroll hard horizontal. Certes, il était un peu rapide (2 octets par balayage), mais il avait le mérite d'être l'un des premiers.
Il est d'ailleurs dommage que le Dr TKC abandonna le CPC après un court passage au Mac Donald's du



Red-TKC Demo.

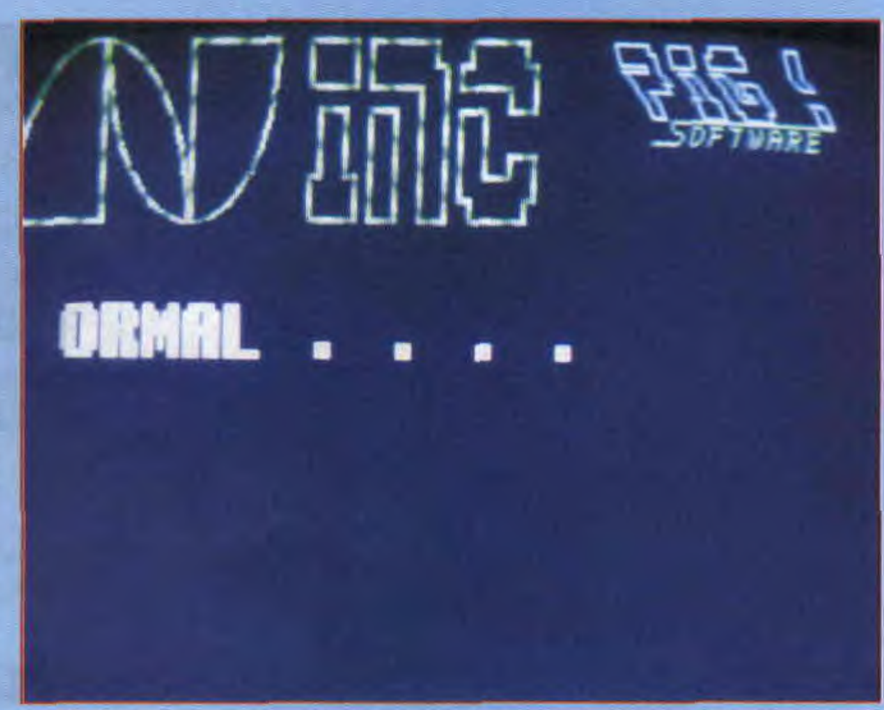
coin, car son travail présageait une nouvelle ère de la démo française.

ET NAQUIT LONGSHOT

Un autre malade des scrolls hard fut Longshot. Ainsi dans « The Longshot Demo », nous avions droit à un superbe scroll hard horizontal, avec une fonte transférée du PC, mais il est vrai que cette démo ne constituait que les prémices, par contre la démo « Revolog » fut, en son temps, une véritable révolution, car elle contenait, tenez-vous bien, un scrolling hard horizontal, un vertical, une fonte PC, des rasters horizontaux (ou encore split-rasters), sans oublier une musique transférée de l'Atari ST.
Notons que notre ami Longshot n'est pas uniquement à l'origine du transfert des musiques ST sur CPC. Ainsi, bien avant les deux démos suscitées et bien avant Titus, il avait utilisé un écran reformaté en overscan dans l'une de ses démos. Il est d'ailleurs amusant de constater que l'overscan, qui est si simple à réaliser sur un pauvre CPC, est une véritable plaie à programmer sur Atari ST, qui est pourtant un ordinateur 16 bits. Enfin, terminons ici ce petit aparté.

LES MALIBUS PROHIBENT

Parallèlement au travail de Longshot (on ne peut pas encore parler de Logon System, car il n'y avait à l'époque qu'un seul membre - comme tout le monde quoi !), apparurent les démos des Malibus Crackers. Bien que ne semblant pas se démarquer des autres petits démomakers, ce groupe nous fit comprendre avec la « Malibu

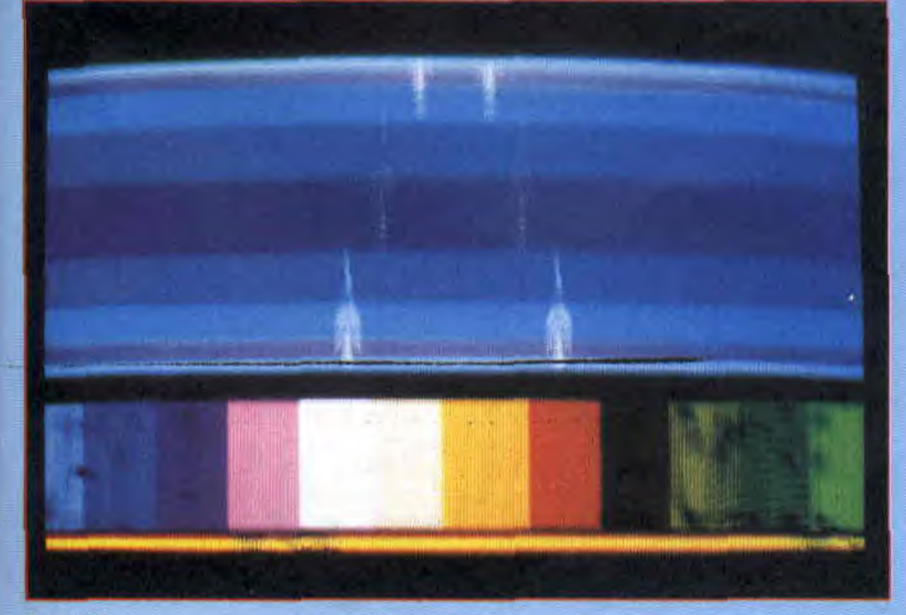


Malibu Demo 4

Demo 4 » qu'il était une valeur sûre (d'ailleurs Naminu, le principal codeur du groupe, rejoindra les rangs de Logon System quelque temps après). Ainsi, on pouvait découvrir un écran overscan avec un scrolling se déplaçant à travers l'écran, accompagné d'un autre scrolling différentiel et enrichi du multimode, j'en passe et des meilleures.

LA MAIN A LA FEFESSE

Egalement à la même époque, un autre programmeur fit parler de lui son nom : Féfesse. Sa première démo vraiment intéressante était une démo avec un scroll en vague. On constate d'ailleurs que le niveau de programmation de Féfesse à l'époque était encore un peu faible, puisque cette démo souffrait de ralentissements lorsqu'on mettait la musique (cela étant, nous sommes restés de bons amis). J'avoue que la plupart des personnes étaient tombées sur le cul en voyant son œuvre, et ce n'est pas notre cher Robby qui me démentira. En effet, LE « pokeman » de cette décennie ne se lassait pas d'écouter la démo musicale de Féfesse et, par la même occasion, d'en faire profiter toute la rédaction d'*Amstrad Cent Pour Cent*. Il s'ensuivit une longue série de démos, puis survint la « Yao Demo ». Des sphères gérant un effet de profondeur, scroll hard horizontal, scroll hard vertical, et même un damier. Voilà en gros ce qui concerne le côté technique de la démo. Jusqu'à présent je ne vous ai pas parlé des textes, mais pour la « Yao » je le ferai rapidement. En effet, cette démo déclencha ce qui sera une guerre de démomakers entre Féfesse et Longshot, (presque aussi violente que celle qui opposera plus tard les démomakers français et leurs camarades allemands). Vous vous doutez bien qu'il s'agit de guéguère de gamins, mais à ce stade un petit pois ressemble étrangement à un ballon de foot.


Revolog.

EN VRAC

Il ne faut pas oublier un autre groupe pionnier de la démo sur CPC : le GPA. Bien que ses démos ne développaient pas forcément de nouvelles techniques, il nous proposait

tout de même un Mad Sinus Scroller qui tournait sur un CPC muni de 64 Ko, alors que celui de P007 ne tournait qu'avec 128 Ko.
Il ne faut pas non plus omettre les Krad'os Crackers qui nous proposèrent, avec leur « Trash Demo », une œuvre graphiquement superbe. M. Zebigboss (le graphiste du groupe) avait réalisé de géniales madballs. La partie programmation était, elle aussi, très bonne, car Mister Plus avait réalisé la démo en overscan, avec un scroll en mode 0, et un writter (bla-bla) en mode 2 utilisant plein de rasters à l'intérieur.

LE DANEMARK REJOINT LA COMMUNAUTE

A cette époque, les deux principaux pays dans la course à la démo étaient la France et l'Allemagne. Mais le Danemark entra dans la course et fut un adversaire avec lequel il fallait compter. Ainsi, par l'intermédiaire de NWC (New Way Cracking), il fit parler de lui. En effet, ce fou nous présenta, avant l'heure, ce qui était à mon avis LA démo sur CPC. Celle-ci nous proposait un véritable remix des meilleures musiques CPC.
En plus, il avait submergé l'écran de tonnes de rasters, tous plus beaux les uns que les autres.
Pour finir, il nous éclatait les yeux avec un scrolling hard horizontal avant tout le monde (à ce que je me rappelle, car je ne peux pas réellement vous dire sa véritable date de création, celles annoncées paraissant légèrement truquées).

DOUCE FRANCE

Revenons un peu à la France. J'évoquais plus haut le moment où Naminu rejoignait le Logon System au détriment des Malibus. Cette association vit le jour, après ce qui fut en son temps une véritable révolution : « The Amazing Demo ». Cette démo, qui remplissait une face entière de disquette, nous présentait des choses d'un type totalement nouveau. En effet, nous nous trouvions face à des démos totalement hard, avec des dizaines de scrolls dans tous les sens, mais aussi (et là ce fut extraordinaire) une musique transférée du ST (bon, je sais, il l'avait déjà fait auparavant, mais attendez la suite nom d'une pipe) qui utilisait de véritables samples. C'est-à-dire qu'en plus des sons gérés par le processeur sonore du CPC, on pouvait entendre parfaitement le bruit des noix de coco s'entrechoquant (pas de sous-entendus). A l'heure actuelle, il est de plus en plus courant d'entendre ce type



Terrific Demo.

de réalisations, car certains codeurs allemands nous ont habitués à agrémenter leurs démos de musique utilisant des samples pour simuler une batterie.

DERNIERE NOUVEAUTE

J'ai dernièrement entendu une musique Amiga sur CPC. Oui, vous avez bien lu, Amiga. Vous devez donc penser que c'est l'œuvre des Logons ou bien des Allemands. Eh bien, détrompez-vous, il s'agit en fait de CJC du CCC qui a réussi cet exploit. Ainsi ce rédacteur de fanzine a réussi ce que les plus grands groupes étaient sur le point de finir (*dixit* ces mêmes groupes) et qui se sont fait grillés à l'arrivée, dommage pour eux.

UNE SUITE TERRIFIANTE

Pour continuer en douceur, naquit : « The Demo ». Elle n'a plus besoin d'être décrite dans ces pages, car elle était à l'époque plébiscitée par tous les journalistes de votre magazine (c'est vrai que les rédacteurs touchaient au passage un petit quelque chose).
Vient ensuite la « Terrific Demo », réalisée par un groupe d'irréductibles Allemands, Thriller, BSC, MCS, Whee... Du beau monde quoi. Cette méga démo nous proposa tout de même quelques idées intéressantes. Par exemple, le menu était non pas formé d'un texte indiquant le nom des démos, mais d'un mini labyrinthe dans lequel il fallait se déplacer pour trouver les portes vers les différentes démos. De plus, dans le genre bonnes idées, Whee nous concocta une partie vraiment sympathique (à mon avis la meilleure de cette démo) qui, bien que jouant une musique originale utilisant des samples de batterie, nous montrait un scrolling horizontal géant dans lequel se trouvaient des dizaines de rasters. Comme je vous l'ai dit plus haut, un MCS en petite forme participa également à cette démo.

LA CHOUCHOU DE POUM

Une autre méga démo qui fit parler d'elle était la « Paradise Demo » du groupe Paradox. En effet, cette démo nous montra un nouvel aspect du démomaking. Ainsi elle prouva que, sans être des bêtes de programmation, on pouvait réaliser quelque chose de très intéressant.
Celle-ci comprenait, entre autres, un menu, réalisé par Gozeur, qui ressemblait plus à une simulation de course à pied qu'à un simple menu. Il fallait choisir sa démo à l'aide d'un chevalier qui se baladait dans un décor assez sympathique, scrollant au fur et mesure de son déplacement. De plus, les parties étaient en général d'un bon niveau. Ainsi, j'aime tout particulièrement celle appellée « Pix Time », qui nous montrait un carré formé de dots (pixels indépendants) bougeant en 3 dimensions. Une autre partie très sympathique était celle intitulée « Artificial Paradise » de Syntax Error du GPA. Cette partie nous présentait un scroll en vague, ainsi qu'un tout aussi élé-



Paradise Demo.

gant scroll en cercle. Notons que certaines parties étaient inspirées d'autres démos (le scroll vertical géant qui ressemble à celui de Fed Crazy dans « The Demo »), mais qui peut prétendre tout découvrir ?

L'ALLEMAGNE RIPOSTE

Une autre démo vraiment délire est la « KKB First », une démo allemande composé de deux parties. La première, qui n'est en fait qu'une intro, nous propose tout de même un écran en overscan, rempli de rasters hyper synchronisés, accompagné d'une sympathique musique. Mais en fait, la vraie prouesse technique reste à venir dans la seconde partie. Celle-ci nous propose un nombre impressionnant de scrollings, de rasters et de musique. Tout cela tenant en 64 Ko. De plus l'esthétique de cette démo a rarement été atteinte et particulièrement chez nos amis allemands, tout comme le côté technique, d'ailleurs.

ETRE EN EXTASE

Un autre allumé nous fit connaître les sommets de l'extase démoniak (vous savez, les sensations jamais ressenties auparavant), j'ai nommé mon ami Overflow. En effet, le créateur de l'une des techniques les plus révolutionnaires sur CPC nous proposa de superbes démos. Ainsi avec la « S & KOH », il nous démontre que le CPC peut bouger deux énormes bobs (sphères), prenant à elles seules toute la largeur de l'écran. Puis, comme si cela ne suffisait pas, il nous assène une partie principale dévastatrice : scroll hard démentiel, logos Logon bougeant sur un décor fixe, et surtout, un superbe rouleau qui fait de magnifiques effets visuels.

LA CREME DES CREMES

Je crois, mais ceci n'engage que moi, qu'à l'heure actuelle la démo la plus impressionnante est la « Face Hugger Ultimate Demo ». Pourquoi donc ? Vous devez penser qu'il s'agit de la démo affichant le plus de scrolls à la fois ou que c'est la démo qui nous frappe le plus les tynpans avec de superbes musiques samplées. Eh bien non, rien de cela, mais simplement ce que l'on n'aurait jamais imaginé voir sur CPC, c'est-à-dire de la véritable 3D. Non pas de la 3D avec de simples dots (ou sphères en français, mais qui, au cas où ce vocable serait étranger à vos chétives intelligences, est synonyme de « bobs »), mais réellement des objets complexes en bobs ou encore des cubes, des vaisseaux tout droit sortis de *la Guerre des étoiles*, et le tout en 3D face pleine, s'il vous plaît.

Megadeth, qui aimerait entendre des musiques plus hard sur CPC

TELECHARGEMENT

Pour finir, je vous propose une liste des démos que vous pourrez trouver sur notre serveur Minitel. Pour cela, tapotez 3615 ACPC et régalez-vous en portant un toast à notre santé. Il est à noter que vous devrez utiliser le kit Télécharge afin de bénéficier de ces démos.

- « Logon Demo 1, 2, 3, 5 » (pourquoi pas de démo 4, me direz-vous ? Simplement parce que personne ne l'a jamais vue, même pas les membres du Logon, si ce n'est Longshot, le codeur de ces démos);
- « The From Beyond I&II » (réalisée par Slash, toujours de chez Logon) ;
- « MCS Demo 4, 5, 6, 7 » (réalisée par MCS) ;
- « The Yao Demo » (réalisée par Féfesse, la partie principale est la plus intéressante);
- « The Malibu Demo 4 » (réalisée par Naminu) ;
- « The Fucking Exams » (réalisée par Fred Crazy & Slash) ;
- « Revolog » (réalisée par Longshot) ;
- « TMS Demo II » (réalisée par le Dr TKC) ;
- « Digit II » (réalisée par Digit) ;
- « Trash Demo » (réalisée par les Krad'os Crackers) ;
- « Remix 1 » (réalisée par NWC) ;
- « S & KOH » (réalisée par Overflow) ;
- « KKB First » (réalisée par les KKB).

Cette liste n'est bien sûr pas exhaustive, et vous retrouverez sur notre serveur bien d'autres bijoux, qu'il ne vous reste plus qu'à découvrir. Allez les démomaniaques, je vous dis au revoir et bonne démonstration.